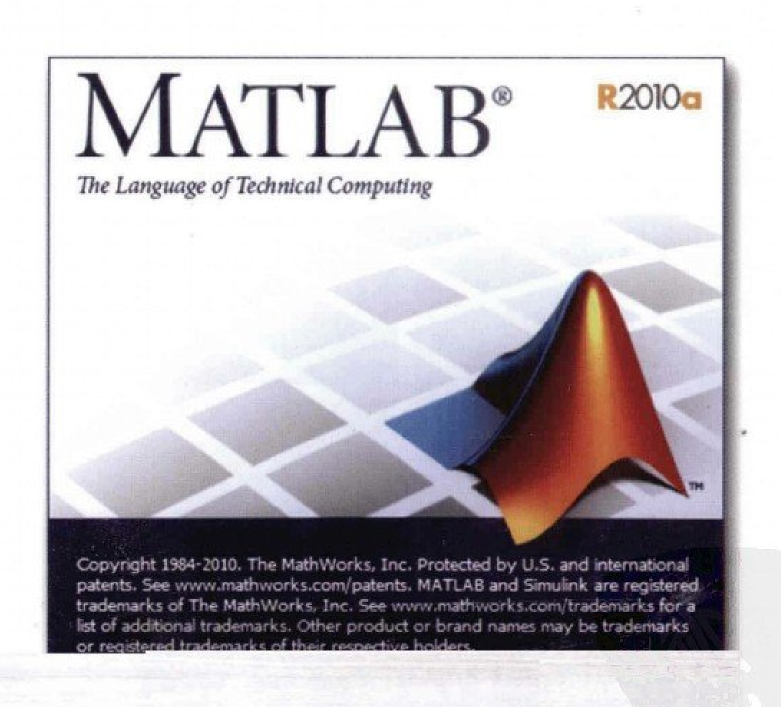
普通高校“十二五”规划教材
PUTONG GAOXIAO SHIERWU GUIHUA JIAOCAI

MATLAB教程

R2010a

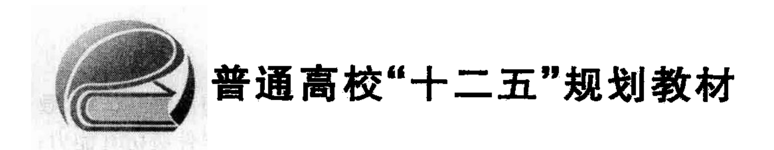
张志涌　杨祖樱　等编著





配有光盘

北京航空航天大学出版社



MATLAB 教程
R2010a

张志涌　杨祖樱　等编著

北京航空航天大学出版社

内 容 简 介

本书以 MATLAB R2010a 为基础修订，系统讲解 MATLAB 基本环境和操作要旨；分章阐述符号计算、数值计算、计算结果可视化及编程精要；举例展现 MATLAB 精华工具 Simulink 的功能级和元器件级仿真能力；举例剖析 MATLAB 界面编辑器的用法和图形用户界面（GUI）的制作要求；简要勾画 MATLAB 和 Word 集成一体的 Notebook 环境。

全书包含 173 个多年凝练的计算范例和 83 个开拓思路的习题。所有算例程序可靠、完整，读者可以完全准确地重现本书所提供的算例结果。书配光盘中附有包含彩色图形的电子版习题答案。书后编有索引，本书所用全部指令及配套的标点符号一览无余。利用索引，读者很容易查阅演示各指令和标点使用方法的节次。

全书由印刷版和电子版结合而成。印刷版便于读者进行系统、全面、长时间连续阅读，便于读者随手翻阅、浏览；而电子版则方便教师制作电子讲稿，方便学生完成电子作业，向读者提供实践本书内容所需的全部可靠程序、色彩信息和动态交互环境，还将随 MATLAB 版本升级而及时地向读者提供新内容。

本书内容充实、篇幅紧凑，是专为理工科院校本科生系统学习 MATLAB 而撰写的，也可供部分研究生使用；既可用于课堂教学教材及课程设计、毕业设计参考用书，也可作为自学用书。

图书在版编目（CIP）数据

MATLAB 教程：R2010a / 张志涌，杨祖樱等编著．--北京：北京航空航天大学出版社，2010．8

ISBN 978－7－5124－0145－7

Ⅰ．①M… Ⅱ．①张…②杨… Ⅲ．①计算机辅助计算－软件包，MATLAB－教材 Ⅳ．①TP391．75

中国版本图书馆 CIP 数据核字（2010）第 130658 号

MATLAB 教程

R2010a

张志涌　杨祖樱　等编著

责任编辑　蔡　喆

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号（邮编 100191）　http://www.buaapress.com.cn

发行部电话：(010)82317024　传真：(010)82328026

读者信箱：bhpress@263.net　邮购电话：(010)82316936

涿州市新华印刷有限公司印装　各地书店经销

*

开本：787×1 092　1/16　印张：22.25　字数：570 千字

2010 年 8 月第 1 版　2010 年 8 月第 1 次印刷　印数：5 000 册

ISBN 978－7－5124－0145－7　定价：43.00 元（含光盘）

前　　言

1. 编写背景

MATLAB(MATrix LABoratory)自1984年问世以来,历经了实践的检验、市场的筛选和时间的凝练,现在已经成为广大科研工作者、高校师生最常用和最可信赖的仿真软件。MATLAB的影响表现在两方面:传统分析方法、设计程式和教材内容在MATLAB平台上可以处理得更为简捷、精确和生动多彩;新的分析方法、设计程式和教材内容正在MATLAB的推动下不断地萌发。

当今欧美高校的理工科教材与MATLAB的关联大致分为三个层面。第一层面,完全不改变原有教材,而把应用MATLAB进行仿真试验的内容另编成册。这个层面的教材最早出现于20世纪80年代的中后期,现在仍有相当一些教材采用这种形式。第二层面,保留或稍许改变原教材内容,而把MATLAB处理教材内容的部分增添成专门的章节。这个层面的教材最早在20世纪90年代初出版。现今,它们已成为国外专业和专业基础教材的主流。第三层面,大幅度地摒弃了原教材中那些在计算尺时代建立起来的分析方法和设计程式,而新建了以MATLAB为基础的现代分析方法和设计程式。这类教材较晚地出现在21世纪初,至今数量也不多。

至于我国,虽然MATLAB在高校已经流传和应用了十余年,但就目前国内理工科专业和专业基础教材而言,包含MATLAB的教材还着实不多见。前不久,我国教育主管部门提出的关于MATLAB作为平台计算软件的指导性意见,必将对我国高等理工科教学的发展产生重要影响,对提高我国后备人才在现代化计算平台上的创新力有十分积极的意义。

2. 编写宗旨

从MATLAB作为高等教学计算平台的基本点出发,融合作者近10余年在本科和研究生MATLAB教学、科研第一线积累的经验,本书编写宗旨定位于:面向本科,立足基础;注重教材内容的稳定,兼顾MATLAB的时代特征。具体措施有:

一、本版新书中将所涉及的数学内容控制在本科教学大纲要求的水平;

二、本书不涉及MATLAB专业工具包(如控制、信号处理、图像处理、通信、金融、生物信息等)的内容,而着力阐述MATLAB如何描述被解算问题、实施计算的指令及多指令相互配用、如何表达或表现计算所得的结果;

三、本书特别强调MATLAB面向复数、面向数组的运算特点,强调向量化编程,与此同时还精心设计了若干算例向读者警示数值计算中的注意事项;

四、考虑到课程设计和毕业设计的需要,本书专辟一章讲授图形用户界面(GUI)的制作要领;

五、本书还展示了依托 MATLAB 建立的 Simulink 的“模块＋鼠标操作”的交互式建模能力，展示了 Simulink 在功能级和元器件级两个层面上的仿真能力。

3. 本书结构

本书由目录、正文、习题、附录、索引和随书光盘组成。

正文共 9 章，包含 173 个算例，83 个习题。章节内容基本上遵循由浅入深原则编排。大量的算例是本教材一大特色。每个算例都经过精心设计，从不同角度展示 MATLAB 的特点、规则和注意事项。习题分章安排在正文之后，习题答案附在光盘中。本教材习题承载两个功能：一，培养学生独立解决问题的能力；二，拓展学生对 MATLAB 的认识。

附录 A 简单介绍字符串数组、胞元数组和构架数组。附录 B 描述光盘的内容和用法。附录 C 的索引中汇集了本教材所涉及的 MATLAB 指令。除标点符号在最前外，一般指令按英文字母次序罗列。每个符号或指令后，列出了本书介绍或使用该指令的具体节次。

随书光盘中 Word 文档(DOC 文件)的主要功用：向教师提供制作本教程电子讲稿的基础文件；向学生提供完成电子作业的模板；向读者提供演练 Notebook 的良好环境。而光盘上的 M，MDL 等文件为读者提供了所有算例在 MATLAB 环境运作必需的文件。

4. 内容介绍

本书共 9 章。

第 1 章　基础准备及入门　详细讲述 MATLAB 运行的基本条件、基本特征和使用方法，讲授如何借助 MATLAB 自带的帮助系统解决所遇到的困难。任何 MATLAB“生手”借助本章都可以比较顺利地跨入 MATLAB 门槛。

第 2 章　符号计算　演绎数学问题的解析计算和任意精度解。该章介绍的解题理念、计算过程、计算结果与高校教科书中的理论内容十分相似，因此学生比较容易接受并应用。此外，在 MATLAB 中，由于符号计算和数值计算采用两个不同的“引擎”，所以本章内容相对独立。

第 3 章　数值数组及向量化运算　阐述数值计算特点、数组运算、向量化编程，以及解决一般数学问题所必需的各种基本函数、逻辑关系表达和基本技法。应该指出，从第 3 章到第 7 章的全部内容是围绕 MATLAB 的主流——数值建模及计算展开的。

第 4 章　数值计算　分类讲述基本数学问题(如微积分、极值、微分方程、矩阵和代数方程、概率统计、多项式和卷积等)的数值计算指令和要领，帮助学生建立正确的数值计算概念。

第 5 章　数据和函数的可视化　阐释理论数学函数可视化的基本步骤、基本指令和协调使用，培养学生借助图形获知离散数据所隐含函数特征的能力。

第 6 章　M 文件和函数句柄　系统介绍 MATLAB 程序中最常用的 4 种控制结构和构建函数调用关系的“函数句柄”。M 文件的意义在于综合使用前几章所介绍的各种基本指令和基本方法，构造较复杂问题的仿真模型，并计算之。

第 7 章　Simulink 交互式仿真集成环境　采用算例引导、纵向深入的方式描述Simulink 模型的交互式创建和仿真方法。列举的 4 个典型算例分别是：基于微分方程的连续系统仿真、基于传递函数的连续系统分析、基于滤波模块的采样离散系统仿真以及基于元器件级模块的

电路瞬态分析。本章无意对 Simulink 解决信号与系统问题、电路分析问题进行全面阐述,而着力于让学生通过典型算例举一反三体验 Simulink 崭新、强大的仿真能力。

第 8 章 第 8 章 图形用户界面(GUI) 借助 4 个典型算例,从入门到纵深,简明地介绍界面编辑器的使用要领,展示典型控件的属性设置和回调函数编写技巧。

第 9 章 Notebook 简要介绍集 Word 与 MATLAB 于一体的 Notebook 工作环境的创建和使用。

5. 教材的时新性和版本的稳定性

软件书籍时新性好坏的标志是软件书籍及时反映升级软件最新变化能力的强弱;而版本稳定性好坏的标志则是阐释该软件的书籍内容在时间推移过程中对于不断升级的软件适配程度的高低。出于“注重稳定兼顾时新”的理念,本教材对内容时新性、版本稳定性的分析和应对措施如下:

(1) 版本稳定性的基础是 MATLAB 核心指令的高度稳定

MATLAB 问世近 30 年,历经大小数十次版本升级,其自身容量已从数百 KB 膨胀到接近 3 GB,其数据结构已从单一类型扩展为多类型,其操作平台已从 DOS 迁移到 Windows,但其基本指令的使用规则、程序流控制却几乎没有任何变化。这完全归功于 MathWorks 公司的远见卓识和精湛的兼容性处理技术。

(2) 教材功能定位有利于增强版本稳定性

本教材既与手册类软件书籍不同,也与专业类软件书籍不同。本教材的功能定位于:阐释软件核心,强调基础应用,注重范例引导,适当展示软件最新功能。

基于这种功能定位,本书除第 1,7,8 章外的其余 6 章所阐释的就是 MATLAB 的核心指令和规则。因此,据 MATLAB 发展史不难预测,这 6 章内容将具有很高的版本稳定性。

(3) 教材阐释的基本原则和程式具有较高的版本稳定性

Simulink 是 MATLAB 中与真实过程(系统)“距离”最近的仿真环境,是 MATLAB 走向实时仿真的最主要途径,是当今 MATLAB 中最具活力、变化最快的工具包。

MATLAB 教材若涉及 Simulink,那么这部分内容就必将是“版本敏感”的,如适配于 MATLAB 6. x 的 Simulink 5. x 中建立的模型在适配于 MATLAB 7. x 的 Simulink 6. x 中就很可能无法运行。但作为 MATLAB 教材,若舍弃 Simulink 内容,将大大局限读者的视野,也大大阻碍学习者对仿真发展趋势的感知。

权衡利弊,本教材第 7 章讲解了 Simulink 的 4 种典型用法。算例的 MDL 文件是在适配于 MATLAB R2010a(即 MATLAB 7. 10)的 Simulink 7. 5 环境中建立的。由于Simulink在版本升级中变化最为频繁,因此这些 MDL 文件有可能在 Simulink 7. 5 的以前或以后版中运行失败。此时,读者通常只要参照本书提供的模型图、建模步骤和参数设置,利用不同 Simulink 环境中的模块重新勾画模型,所得的 MDL 就可以在相应的Simulink环境中成功运行。

(4) 逐年修订保障教材的时新性

“教材时新”是本书作者和出版社尽心保障的另一个特点。这基于两方面的考虑:一,2006 年 MATLAB 的制造商宣布,MATLAB 将每隔半年升级一次;二,教材的时新性有利于增强读者、学生对 MATLAB 的“亲和感”和“学习心态的愉悦”。

本书作者和出版社将通力合作，对本教材进行逐年修订，使印刷版确保“时新”，而电子版又向前兼容。换句话说，本书的印刷版将随 MATLAB 的升级不断修订，而随书光盘将提供与以前各版适配的相关内容。

6. 教学建议

(1) 教学环境和形式

- 本教材内容适于在多媒体教室讲授。本教材中所有算例的计算结果（包括数据和图形）都应该在教学现场实时产生，以便学生亲眼目睹教师操作，感受计算过程和计算结果。
- 对于涉及 MATLAB 内容较多的课程，不宜采用 PowerPoint 写成的 PPT 幻灯片作为电子讲稿。本书作者建议采用“Word + MATLAB”构成的 M－book 模板编写电子讲稿。这样，通过本课程的潜移默化，学生更容易掌握 M－book 的使用。光盘中的 Word 文档（DOC 文件）可方便于教师制作电子讲稿。
- 学习本教材的每个学生都应该在计算机上亲自演练本教材中的算例。要特别重视算例指令的直接键入练习，只有这样才能加深对 MATLAB 的理解，纠正自己的误解和误操作。作者建议：学生采用光盘上的 M－book 模板解答本书习题。

(2) 教学内容安排

- 作为入门内容的第 1 章必须最先讲授，但不必太细。除 MATLAB 及其工作界面的最基本特点和操作技法外，其余详细内容可以渗透在本教材以后的课程中介绍。
- 本书第 9 章内容不必使用单独的课时讲授。假若教师课堂教学采用 M－book 写成的讲稿在 Notebook 环境中进行，那么第 9 章内容将在教学过程中得到最好的传授。
- 本教材之所以把“符号计算”安排在第 2 章，是考虑到本章所涉及计算推演模式相似于大学（数学）教材。假如摒弃以上考虑，那么本章内容也可以安排在第 3、第 4 章以后讲授。
- 假如教学课时有限，那么下列节次可以酌情少讲或不讲：第 2.7 节、第 2.8.2 节、第 4.3 节、第 4.4.2 节、第 5.3.4 节、第 5.4 节、第 7 章。

7. 致　谢

无论在本教材的讲稿阶段，还是在成书阶段，得到了靳种宝、李娟娟、胡丽珍、谢逢博、冯子豪、王贵银等研究生的帮助和支持，也得到南京邮电大学各级领导的鼓励和关心。借本书出版之际，作者向他们表示真诚的感谢。

最后还要感谢北京航空航天出版社的同志们，是他们一再鼓励作者结合讲稿和读者的反馈意见对前书《MATLAB 教程》进行修订、改编。

本书基本内容虽经多年教学的筛选提炼，但限于作者知识水平，书中可能仍有疏漏、错误和偏见，在此，作者恳切期望得到各方面专家和广大读者的指教。作者 E-mail：zyzh@njupt.edu.cn。

作　者

2006 年 4 月初成于南京

2010 年 6 月修改于南京

2010a 适配版修订说明

在本书 2010a 适配版中，进行了如下重要修订：

（1）根据 MATLAB R2010a 版本，第 1 章中的所有界面及相应的解释都已更新；由于 MATLAB 自身帮助系统的变化，本书第 1.9 节“帮助系统及其使用”是全新的。

（2）从 MATLAB R2008b 开始，符号计算的默认计算引擎就不再使用 Maple，而使用 MuPAD。为适应这种变化，本书第 2 章内容已经全部重写。为使长期使用本书的读者不至于产生生疏感，重写时尽量保持了原先的章节框架和例题内容。

（3）随 MATLAB 的升级，Simulink 库模块功能得到不断的完善和扩充。为反映这种变化，本书第 7 章在保持原先例题要求基本不变的前提下进行了大幅度地修订。

（4）为满足课程设计、毕业设计的需要，本书新增第 8 章　图形用户界面(GUI)。

作　者

2010 年 6 月于南京

出版说明

根据 The MathWorks 公司公布的最新版本 MATLAB R2010a，本书在《MATLAB 教程 R2008a》的基础上对印刷版内容和光盘内容进行了修改和调整。

目前，本书除完全适配于 MATLAB R2010a 以外，在配套光盘中还保留有适配于 MATLAB R2006a 至 2008a 版本的电子文档，以供未能及时升级软件的读者参考使用。

本书是作者定位于高校教材，以基础、实用的内容和精简、合理的篇幅为特色，修订出版的又一力作。自出版以来得到了业内同仁和广大读者的支持和关注。为了不辜负大家的希望，我们将努力做好随软件版本升级的图书更新工作，并不断充实、完善本书的内容。

恳请各位教师、专家和广大读者为本书提出更多意见和建议！本书责任编辑联系电话：010－82317036，E-mail：zhecai@263.net。

北京航空航天大学出版社

2010 年 7 月

目　　录

第1章 基础准备及入门

本章有三个主要内容：一是讲述 MATLAB 正常运行所必备的基础条件；二是简明地介绍 MATLAB 及其操作桌面 Desktop 的基本使用方法；三是全面介绍 MATLAB 的帮助系统。

本章的第 1.1，1.2 节讲述 MATLAB 的正确安装方法和 MATLAB 环境的启动。因为指令窗是 MATLAB 最重要的操作界面，所以本章用第 1.3，1.4 两节以最简单通俗的叙述、算例讲述指令窗的基本操作方法和规则。这部分内容对 MATLAB 的几乎各种版本都适用。第 1.5～1.8 节专门介绍 MATLAB 最常用的 5 个交互界面：历史指令窗、当前目录浏览器、工作空间浏览器、变量编辑器、M 文件编辑器。鉴于实际应用中，帮助信息和求助技能的重要性，本章第 1.9 节专门讲解 MATLAB 的帮助体系和求助方法。

作者建议：不管读者此前是否使用过 MATLAB，都不要忽略本章。

1.1 MATLAB 的安装和工具包选择

MATLAB 只有在适当的外部环境中才能正常运行。因此，恰当地配置外部系统是保证 MATLAB 运行良好的先决条件。MATLAB 本身可适应于许多机种和系统，如 PC 机和 Unix 工作站等。但本节只针对我国使用最广的 PC 机系统给予介绍。

PC 机用户常常需要自己安装 MATLAB。MATLAB R2010a（即 MATLAB 7.10）版要求 Windows 2000 或 Windows XP 平台。下面介绍从光盘上安装 MATLAB 的方法。

一般说来，当 MATLAB 光盘插入光驱后，会自动启动"安装向导"。假如自动启动没有实现，那么可以在＜我的电脑＞或＜资源管理器＞中找到并双击 setup.exe 应用程序，使"安装向导"启动。安装过程中出现的所有界面都是标准的，用户只要按照屏幕提示操作，如输入用户名、单位名、口令等就行。

在安装 MATLAB R2010a 版时，会出现一个界面，该界面上有两个选项：Typical 和 Custom。由于近年计算机硬盘容量通常很大，所以一般为使用方便，可直接点选"Typical"。

安装完成后，一般会产生两个目录：

① MATLAB 软件所在的目录

- 该目录位置及目录名，都是用户在安装过程中指定的。比如，C:\MATLAB R2010a。
- 该目录包含 MATLAB 运行所需的所有文件，如启动文件、各种工具包等。

② MATLAB 自动生成的供用户使用的工作目录

- 该目录是安装 MATLAB 时自动生成的，专供用户存放操作 MATLAB 中产生的中间

文件。

- 该工作目录的名称是"MATLAB",一般登录在 C:\Documents and Settings\acer\My Documents 文件夹下。(注意:文件夹名中的"acer"会随电脑不同而变。)
- 该工作目录 C:\Documents and Settings\acer\My Documents\MATLAB 被自动记录在 MATLAB 的搜索路径中。因此,在该目录上的 M 文件、MAT 文件、MDL 文件等都能被 MATLAB 搜索到。

1.2 Desktop 操作桌面的启动

1.2.1 MATLAB 的启动

(1) 方法一

当 MATLAB 安装到硬盘上以后,一般会在 Windows 桌面上自动生成 MATLAB 快捷方式图标。在这种情况下,只要直接双击图标即可启动 MATLAB,打开如图 1.2-1 所示的 MATLAB 操作桌面(Desktop)。注意:本书作者建议用户优先采用"方法一"。

(2) 方法二

假如 Windows 桌面上没有 MATLAB 图标,那么可以双击 matlab 文件夹下的快捷方式图标MATLAB。

1.2.2 Desktop 操作桌面简介

MATLAB R2010a 版的 Desktop 操作桌面是一个高度集成的 MATLAB 工作界面。其默认形式如图 1.2-1 所示。该桌面的上层铺放着三个最常用的界面:指令窗(Command Window)、当前目录(Current Directory)浏览器、MATLAB 工作内存空间(Workspace)浏览器、历史指令(Command History)窗。

(1) 指令窗

该窗是进行各种 MATLAB 操作的最主要窗口。在该窗内,可键人各种送给 MATLAB 运作的指令、函数、表达式;显示除图形外的所有运算结果;运行错误时,给出相关的出错提示。

(2) 当前目录浏览器

在该浏览器中,展示着子目录、M 文件、MAT 文件和 MDL 文件等。对该界面上的 M 文件,可直接进行复制、编辑和运行;界面上的 MAT 数据文件,可直接送入 MATLAB 工作内存。对该界面上的子目录,可进行 Windows 平台的各种标准操作。此外,在当前目录浏览器正下方,还有一个"文件概况窗"。该窗显示所选文件的概况信息。比如该窗会展示:M 函数文件的 H1 行内容,最基本的函数格式,所包含的内嵌函数和其他子函数。

(3) 工作空间浏览器

该浏览器默认地位于当前目录浏览器的后台。该窗口罗列出 MATLAB 工作空间中所有的变量名、大小、字节数,在该窗中,可对变量进行观察、图示、编辑、提取和保存。

(4) 历史指令窗

该窗记录已经运作过的指令、函数、表达式,及它们运行的日期、时间。该窗中的所有指令、文字都允许复制、重运行及用于产生 M 文件。

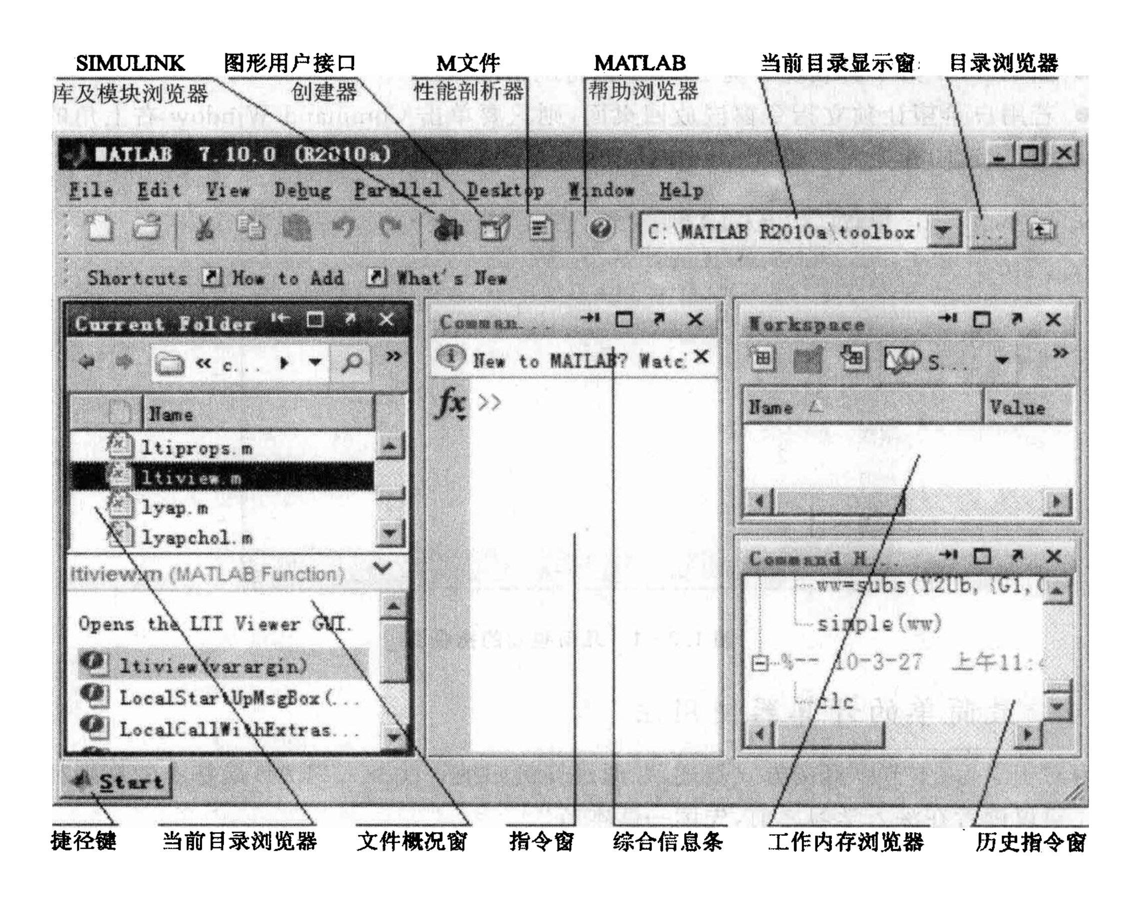


图 1.2-1 Desktop 操作桌面的默认形式

(5) 捷径(Start)键

引出通往本 MATLAB 所包含的各种组件、模块库、图形用户界面、帮助分类目录、演示算例等的捷径,并向用户提供自建快捷操作的环境。

1.3 Command Window 运行入门

MATLAB 的使用方法和界面有多种形式。但最基本的,也是入门时首先要掌握的是:MATLAB 指令窗(Command Window)的基本表现形态和操作方式。通过本节的学习,读者将对 MATLAB 使用方法有一个良好的初始感受。

1.3.1 Command Window 指令窗简介

MATLAB 指令窗默认地位于 MATLAB 桌面的中间(见图 1.2-1)。假如,用户希望得到脱离操作桌面的几何独立指令窗,只要单击该指令窗右上角的■键,就可获得如图 1.3-1 所示的指令窗。

说明

- 图 1.3－1 指令窗表现了例 1.3－1 运行的情况。
- 若用户希望让独立指令窗嵌放回桌面，则只要单击 Command Window 右上角的 ↘ 按钮，或选中指令窗菜单{Desktop:Dock Command Window}便可。

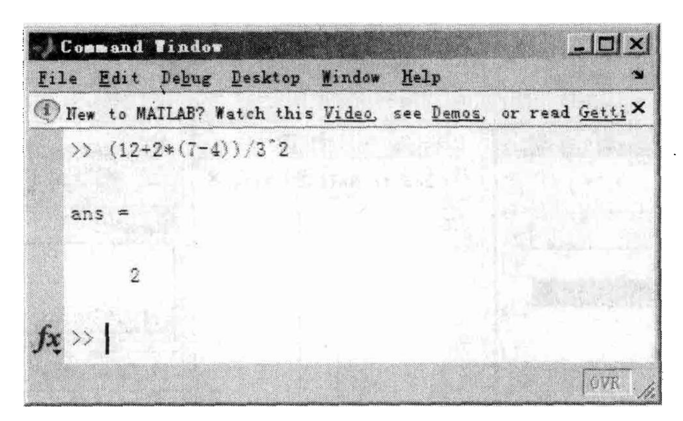


图 1.3－1　几何独立的指令窗

1.3.2　最简单的计算器使用法

为易于学习，本节以算例方式叙述，并通过算例归纳一些 MATLAB 最基本的规则和语法结构。建议读者在深入学习之前，先读一读本节。

例【1.3－1】　求$[12+2\times(7-4)]\div 3^2$的算术运算结果。本例演示：最初步的指令输入形式和必需的操作步骤。

（1）用键盘在 MATLAB 指令窗中输入以下内容：

```
>>(12 + 2 * (7 - 4))/3^2
```

（2）在上述表达式输入完成后，按［Enter］键，该指令被执行，并显示如下结果：

```
ans =

     2
```

说明

- 本例在指令窗中实际运行的情况参见图 1.3－1。
- 指令行头首的“>>”是指令输入提示符，它是自动生成的。本书在此后的输入指令前将不再带提示符“>>”。理由是：第一，为使表达简捷；第二，本书用 MATLAB 的 M－book写成，而在 M－book 中运行的指令前是没有提示符的。
- MATLAB 的运算符（如＋，－等）都是各种计算程序中常见的习惯符号。
- 一条指令输入结束后，必须按［Enter］键，该指令才被执行。
- 由于本例输入指令是不含赋值号的表达式，所以计算结果被赋给 MATLAB 的一个默认变量 ans（英文 answer 的缩写）。

例【1.3－2】　指令的续行输入。本例演示：由于指令太长，或出于某种需要，输入指令行必须多行书写时，该如何处理。

```
S = 1 - 1/2 + 1/3 - 1/4 + ...
1/5 - 1/6 + 1/7 - 1/8

S =
    0.6345
```

说明

- MATLAB 用 3 个或 3 个以上的连续黑点表示续行，即表示下一行是上一行的继续。
- 本例指令中包含赋值号，因此表达式的计算结果被赋给了变量 S。
- 指令执行后，变量 S 被保存在 MATLAB 的工作空间（Workspace）中备用。如果用户不用 clear 指令清除它，或对它重新赋值，那么该变量会一直保存在工作空间中，直到本 MATLAB 指令窗被关闭为止。

1.3.3 数值、变量和表达式

1.3.2 节的算例只演示了计算器功能，那仅是 MATLAB 全部功能中小小的一角。为深入学习 MATLAB，有必要系统地介绍一些基本规定。本节先介绍关于变量的若干规定。

1. 数值的记述

MATLAB 的数值采用习惯的十进制表示，可以带小数点或负号。以下记述都合法：

```
3     -99     0.001     9.456     1.3e-3     4.5e33
```

在采用 IEEE 浮点算法的计算机上，数值通常采用“占用 64 位内存的双精度”表示。其相对精度是 eps（MATLAB 的一个预定义变量），大约保持有效数字 16 位。数值范围大致为 10^{-308}～10^{308}之间。

2. 变量命名规则

- 变量名、函数名是对字母大小写敏感的。如变量 myvar 和 MyVar 表示两个不同的变量。sin 是 MATLAB 定义的正弦函数名，但 SIN，Sin 等都不是。
- 变量名的第一个字符必须是英文字母，最多可包含 63 个字符（英文、数字和下连符）。如 myvar201 是合法的变量名。
- 变量名中不得包含空格、标点、运算符，但可以包含下连符。如变量名 my_var_201 是合法的，且读起来更方便。而 my, var201 由于逗号的分隔，就不能表示一个变量名。

3. MATLAB 默认的数学常数

MATLAB 为一些数学常数（Math Contants）预定义了变量名，见表 1.3-1。MATLAB 每次启动，都产生这些变量。这些变量都有特殊含义和用途。建议用户在编写指令和程序时，尽可能不对表 1.3-1 所列预定义变量名重新赋值，以免产生混淆。

表 1.3-1 MATLAB 为数学常数预定义的变量名

预定义变量	含 义	预定义变量	含 义
eps	浮点数相对精度 2^{-52}	NaN 或 nan	不是一个数(Not a Number),如 0/0,∞/∞
i 或 j	虚单元		
Inf 或 inf	无穷大,如 1/0	pi	圆周率 π
intmax	可表达的最大正整数,默认(2147483647)	realmax	最大正实数,默认 1.7977×10^{308}
intmin	可表达的最小负整数,默认(-2147483648)	realmin	最小正实数,默认 $2.2251e\times10^{-308}$

说明

- 假如用户对表中任何一个预定义变量中进行赋值,则该变量的默认值将被用户新赋的值临时覆盖。所谓"临时"是指:假如使用 clear 指令清除 MATLAB 内存中的变量,或 MATLAB 指令窗被关闭后重新启动,那么所有的预定义变量将被重置为默认值,不管这些预定义变量曾被用户赋过什么值。
- 在遵循 IEEE 算法规则的机器上,被 0 除是允许的。它不会导致程序执行的中断,只是在给出警告信息的同时,用一个特殊名称(如 Inf 或 NaN)记述。这个特殊名称将在以后的计算中以合理的形式发挥作用。
- 关于它们的更详细的帮助信息,可在 MATLAB 帮助浏览器左侧 Contents 页的 <MATLAB/Functions/Mathematics/Math Contants>找到。

例【1.3-3】 运用以下指令,以便初步了解预定义变量。本例演示:各常数的含义。

```
format short e
RMAd = realmax('double')          % 双精度类型(默认)时最大实数
RMAs = realmax('single')          % 单精度类型时最大实数
RMAd =
  1.7977e+308
RMAs =
  3.4028e+038

IMA64 = intmax('int64')           % int64 整数类型时最大正整数
IMA32 = intmax                    % int32(默认)整数类型时最大正整数
IMA32 = intmax('int16')           % int16 整数类型时最大正整数
IMA64 =
  9223372036854775807
IMA32 =
  2147483647
IMA32 =
  32767
```

```
e1 = eps                          % 双精度类型时的相对精度
e2 = eps(2)                       % 表达 2 时的绝对精度
e1 =
    2.220446049250313e-016
e2 =
    4.440892098500626e-016

pi
ans =
  3.141592653589793
```

4. 运算符和表达式

(1) 经典教科书上的算术运算符在 MATLAB 中的表达方式如表 1.3-2 所列。

表 1.3-2 MATLAB 表达式的基本运算符

类 别	数学表达式	矩阵运算符	数组运算符
加	$a+b$	a + b	a + b
减	$a-b$	a - b	a - b
乘	$a\times b$	a * b	a .* b
除	$a\div b$	a / b 或 b \ a	a ./ b 或 b .\ a
幂	a^b	a ^ b	a .^ b
圆括号	( )	( )	( )

说明

- MATLAB 面向复数设计，其所有运算定义在复数域上，所以对于方根问题，运算默认只返还一个主解。要得到复数的全部解，必须专门编写程序(参见例 1.3-6)。
- MATLAB 面向矩阵/数组设计，标量被看作(1×1)的矩阵/数组。
- 数组运算的"乘、除、幂"规则与相应矩阵运算根本不同。前者的算符比后者多一个"小黑点"(参见例 1.3-9，例 1.3-10。更详细说明请看第 3 章)。
- MATLAB 用左斜杠或右斜杠分别表示"左除"或"右除"运算。对标量而言，"左除"和"右除"的作用结果相同。但对矩阵来说，"左除"和"右除"将产生不同的结果。
- 关于它们的更详细的帮助信息，可在 MATLAB 帮助浏览器左侧 Contents 页的 <MATLAB/ User Guide/ Programming Fundamentals/ Basic Program Components/ Operators/ Arithmetic Operations>找到。
- 关于它们的帮助信息，也可在 MATLAB 帮助浏览器左上方的搜索栏中输入"Arithmetic Operations"经搜索获得。

(2) MATLAB 书写表达式的规则与"手写算式"几乎完全相同。

- 表达式由变量名、运算符和函数名组成。
- 表达式将按常规的优先级自左至右执行运算。

- 优先级的基本规定是：指数运算级别最高，乘除运算次之，加减运算级别最低。
- 括号可以改变运算的次序。
- 书写表达式时，赋值符“＝”和运算符两侧允许有空格，以增加可读性。

5. 面向复数设计的运算——MATLAB特点之一

MATLAB的所有运算都是定义在复数域上的。这样设计的好处是：在进行运算时，不必像其他程序语言那样把实部、虚部分开处理。为描述复数，虚数单位用预定义变量 i 或 j 表示。

复数 $z=a+b\mathrm{i}=re^{\mathrm{i}\theta}$ 直角坐标表示和极坐标表示之间转换的MATLAB指令如下：

real(z)　　给出复数 z 的实部 $a=r\cos\theta$

imag(z)　　给出复数 z 的虚部 $b=r\sin\theta$

abs(z)　　给出复数 z 的模 $\sqrt{a^2+b^2}$

angle(z)　　以弧度为单位给出复数 z 的幅角 $\arctan\frac{b}{a}$

例【1.3-4】 复数 $z_1=4+3\mathrm{i}$，$z_2=1+2\mathrm{i}$，$z_3=2e^{\frac{\pi}{6}\mathrm{i}}$ 表达，及计算 $z=\frac{z_1z_2}{z_3}$。本例演示：正确的复数输入法；涉及复数表示方式的基本指令。

（1）经典教科书的直角坐标表示法

```
z1 = 4 + 3i                          % 合法,但建议少用或不用
z1 =
   4.0000 + 3.0000i
```

说明

- 建议不要使用这种输入格式。因为这种书写格式，只适用于“数值标量”复数，而不适用于“数值矩阵”。
- 在这种书写格式中，3i 是一个完整的虚数，在 3 和 i 之间不许空格存在。

（2）采用运算符构成的直角坐标表示法和极坐标表示法

```
z2 = 1 + 2 * i                       % 运算符构成的直角坐标表示法
z3 = 2 * exp(i * pi/6)               % 运算符构成的极坐标表示法
z = z1 * z2/z3
z2 =
   1.0000 + 2.0000i
z3 =
   1.7321 + 1.0000i
z =
   1.8840 + 5.2631i
```

（3）复数的实部、虚部、模和幅角计算

```
real z = real(z)
image z = imag(z)
magnitude z = abs(z)
angle z radian = angle(z)                % 弧度单位
angle z degree = angle(z) * 180/pi       % 度数单位
real_z =
```

```
     1.8840
image_z =
     5.2631
magnitude_z =
     5.5902
angle_z_radian =
     1.2271
angle_z_degree =
   70.3048
```

【例 1.3-5】 图示复数 $z_1=4+3i$，$z_2=1+2i$ 的和(见图 1.3-2)。本例演示：MATLAB 的运算在复数域上进行；指令后"分号"的作用；复数加法的几何意义；展示 MATLAB 的可视化能力(让读者感受，但不要求理解)。

```
z1 = 4 + 3 * i;z2 = 1 + 2 * i;              % 在一个物理行中，允许输入多条指令
                                            % 同一行各指令间要用"分号"或"逗号"分开
                                            % 指令后采用"分号"，使运算结果不显示
z12 = z1 + z2
% 以下用于绘图
clf,hold on                                 % clf 清空图形窗。逗号用来分隔两个指令
plot([0,z1,z12],'-b','LineWidth',3)
plot([0,z12],'-r','LineWidth',3)
plot([z1,z12],'ob','MarkerSize',8)
hold off,grid on,
axis equal
axis([0,6,0,6])
text(3.5,2.3,'z1')
text(5,4.5,'z2')
text(2.5,3.5,'z12')
xlabel('real')
ylabel('image')
shg

z12 =
   5.0000 + 5.0000i
```

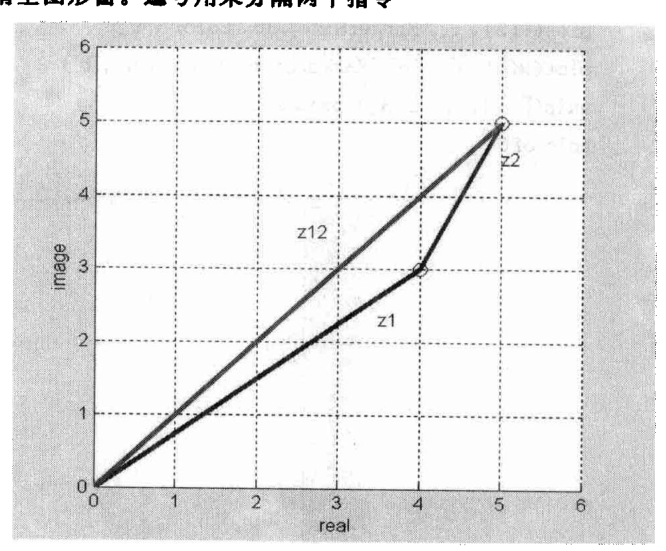


图 1.3-2　两个复数相加

【例 1.3-6】 用 MATLAB 计算 $\sqrt[3]{-8}$ 能得到 -2 吗？本例演示：MATLAB运算定义在复数域的实质；指令后"分号"抑制运算结果的显示；MATLAB 的方根运算规则；更复杂指令的表示方式；展现 MATLAB 的图形表现力(见图 1.3-3)。(对于本例指令，读者体验即可，不必强求理解。)

(1) 直接计算时得到处于第一象限的方根

```
a = -8;
r_a = a^(1/3)        % 求 3 次方根
r_a =
```

```
   1.0000 + 1.7321i
```

(2) $\sqrt[3]{-8}$的全部方根计算

```
% 先构造一个多项式 p(r) = r^3 - a
p = [1,0,0, - a];          % p 是多项式 p(r)的系数向量
                           % 指令末尾的"英文状态分号"使该指令运行后,不显示结果
R = roots(p)               % 求多项式的根
R =
  - 2.0000
   1.0000 + 1.7321i
   1.0000 - 1.7321i
```

(3) 图形表示(见图 1.3-3)

```
MR = abs(R(1));             % 计算复根的模
t = 0:pi/20:2 * pi;         % 产生参变量在 0 到 2 * pi 间的一组采样点
x = MR * sin(t);
y = MR * cos(t);
plot(x,y,'b:'),grid on                % 画一个半径为 R 的圆
                                      % 注意"英文状态逗号"在不同位置的作用
hold on
plot(R(2),'.','MarkerSize',30,'Color','r')        % 画第一象限的方根
plot(R([1,3]),'o','MarkerSize',15,'Color','b')    % 画另两个方根
axis([ - 3,3, - 3,3]),axis square                  % 保证屏幕显示呈真圆
hold off
```

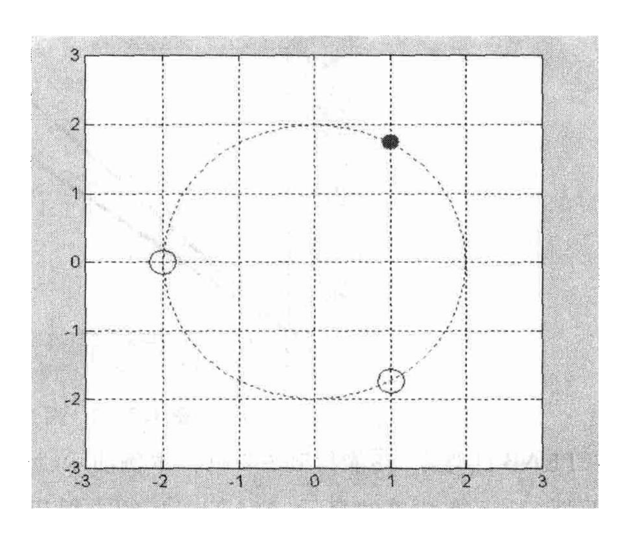


图 1.3-3 -8 的全部三次方根分布

说明

- 本例有助于理解 MATLAB 的计算特点。
- 对复数进行方根运算时,MATLAB 只给出处于"第一象限"的那个根。

6. 面向数组设计的运算——MATLAB 特点之二

在 MATLAB 中，标量数据被看作(1×1)的数组(Array)数据。所有的数据都被存放在适当大小的数组中。为加快计算速度(即运算的向量化处理)，MATLAB 对以数组形式存储的数据设计了两种基本运算：一种是所谓的数组运算；另一种是所谓的矩阵运算。在此仅以算例展示 MATLAB 的计算特点，更详细的叙述请参见第 3 章。

例【1.3-7】 实数数组 $\boldsymbol{AR}=\begin{bmatrix}1 & 3\\2 & 4\end{bmatrix}$的“一行”输入法。本例演示：二维数组的最基本、最常用输入法；二维数组输入的三大要素。

(1) 在键盘上输入下列内容

```
AR= [1,3;2,4]
```

(2) 按 [Enter]键，指令被执行

(3) 在指令执行后，MATLAB 指令窗中将显示以下结果

```
AR =
     1     3
     2     4
```

说明

- 在 MATLAB 中，不必事先对数组维数及大小做任何说明，内存将自动配置。
- 二维数组输入的三大要素：数组标识符“[ ]”；元素分隔符空格或逗号“,”；数组行间分隔符分号“;”或“回车”。注意：所有标点符号都是“英文符号”。
- MATLAB 对字母大小写是敏感的。比如本例中的数组赋给了变量 AR，而不是 Ar，aR，或 ar。
- 在全部键入一个指令行内容后，必须按下 [Enter]键，该指令才会被执行。请读者务必记住此点。出于叙述简明的考虑，本书此后将不再重复提及此操作。

例【1.3-8】 实数数组 $\boldsymbol{AI}=\begin{bmatrix}5 & 7\\6 & 8\end{bmatrix}$的“分行”输入法。

```
AI=[5,7
    6,8]
AI =
     5     7
     6     8
```

说明

- 本例采用这种输入法是为了视觉习惯。当然，对于较大的数组也可采用此法。
- 在这种输入方法中，“回车”用来分隔数组中的行。

例【1.3-9】 对复数数组 $\boldsymbol{A}=\begin{bmatrix}1-5i & 3-7i\\2-6i & 4-8i\end{bmatrix}$进行求实部、虚部、模和幅角的运算。本例演示：复数数组的生成；MATLAB 指令对数组元素“并行操作”的实质。

(1) 创建复数数组

```
AR=[1,3;2,4];AI=[5,7;6,8];
```

```
A = AR - AI * i                              %形成复数数组
A =
   1.0000 - 5.0000i    3.0000 - 7.0000i
   2.0000 - 6.0000i    4.0000 - 8.0000i
```

(2) 求复数数组的实部和虚部

```
A real = real(A)
A image = imag(A)
A_real =
     1     3
     2     4
A_image =
    -5    -7
    -6    -8
```

(3) 求复数矩阵中各元素的模和幅角——循环法(笨拙的方法)

```
for m = 1:2
    for n = 1:2
        Am1(m,n) = abs(A(m,n));
        Aa1(m,n) = angle(A(m,n)) * 180/pi;      % 以度为单位计算幅角
    end
end
Am1,Aa1
Am1 =
    5.0990    7.6158
    6.3246    8.9443
Aa1 =
  -78.6901  -66.8014
  -71.5651  -63.4349
```

(4) 求复数矩阵中各元素的模和幅角——直接法

```
Am2 = abs(A)
Aa2 = angle(A) * 180/pi
Am2 =
    5.0990    7.6158
    6.3246    8.9443
Aa2 =
  -78.6901  -66.8014
  -71.5651  -63.4349
```

说明

- 函数 real、imag、abs、angle 是同时、并行地作用于数组的每个元素的。对 4 个元素运算所需的时间大致与对单个元素所需时间相同。这有利于运算速度的提高。这是“向量化”运算的一种形式。
- 本例给出了循环法求各元素模和幅角的指令。这不是很有效的计算方法。对于 MATLAB 以外的许多编程语言来说,可能不得不采用“循环”处理方式来解本例。记

住：对于 MATLAB 来说，应该尽量摒弃“循环”处理，而采用“向量化”处理方式。

例【1.3－10】 画出衰减振荡曲线 $y=e^{-\frac{t}{3}}\sin 3t$，$t$ 的取值范围是 $[0,4\pi]$（见图 1.3－4）。本例演示：展示数组运算的优点；展示 MATLAB 的可视化能力。

```
t = 0:pi/50:4 * pi;                 % 定义自变量 t 的取值数组
y = exp( - t/3). * sin(3 * t);      % 计算与自变量相应的 y 数组，注意：乘法符前面的小黑点
plot(t,y,'-r','LineWidth',2)        % 绘制曲线
axis([0,4 * pi, - 1,1])
xlabel('t'),ylabel('y')
```

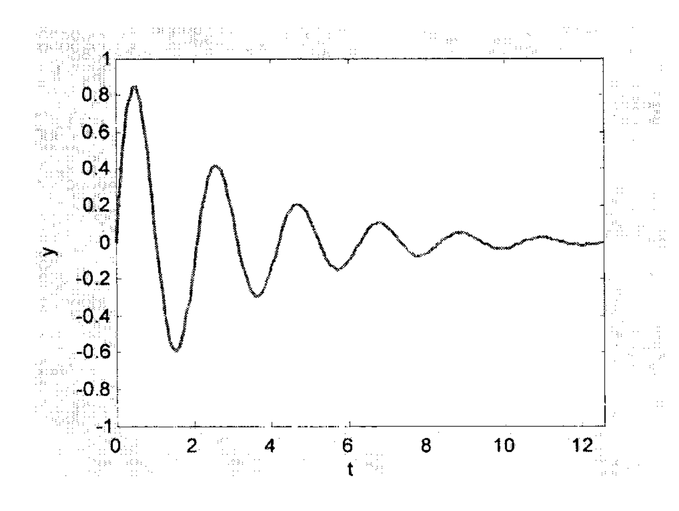


图 1.3－4　衰减振荡曲线

说明

- 本例第二条指令中的“. * ”符号表示乘法是在两个数组相同位置上的元素间进行的。本书把这种乘法称为“数组乘”。数组乘的引入，不但使得程序简洁自然，而且避免了耗费机时的“循环计算”。关于数组运算的详细叙述请参见第 3 章。
- 本例第二条指令是典型的“向量化”处理形式。本书作者建议，只要可能，应尽量采用“向量化”运算形式。

例【1.3－11】 复数矩阵 $\boldsymbol{B}=\begin{bmatrix}3+2i & 2+6i\\5+3i & 4-2i\end{bmatrix}$ 的生成，及计算 $\boldsymbol{A}\cdot\boldsymbol{B}$ 矩阵乘积（$\boldsymbol{A}$ 取自例 1.3－9）。本例演示：MATLAB 矩阵运算指令的简捷性。

```
B = [3 + 2i,2 + 6i;5 + 3 * i,4 - 2 * i]        % 复数数组的又一种输入方式
                                               % 注意标点符号的作用
C = A * B                                      % 矩阵乘法
B =
   3.0000 + 2.0000i    2.0000 + 6.0000i
   5.0000 + 3.0000i    4.0000 - 2.0000i
C =
  49.0000 - 39.0000i  30.0000 - 38.0000i
  62.0000 - 42.0000i  40.0000 - 40.0000i
```

说明

- 当数组被赋予“变换”属性时，二维数组就被称为矩阵。只有当两个矩阵的“内维大小相等”时，矩阵乘法才能进行。本例中，矩阵 ***A*** 的列数与矩阵 ***B*** 的行数相等，所以可以进行 ***A*** 乘 ***B*** 运算。
- 从表达方式看，“矩阵相乘”的指令格式与“标量相乘”指令格式一样。在其他编程语言中，矩阵乘法不得不依赖“循环”进行。
- MATLAB 之所以能把矩阵运算表达得像标准“线性代数”那样简洁易读、自然流畅，是由于 MATLAB 的设计者采用了“面向对象”编程技术。

1.4 Command Window 操作要旨

前一节借助算例，使读者对 MATLAB 指令窗的使用方法有了一个直观的感受。本节将在上节的基础上对控制指令窗的指令和操作进行较系统的归纳，以便读者更全面地了解 MATLAB，更方便地使用 MATLAB。

1.4.1 指令窗的显示方式

1. 默认的输入显示方式

从 MATLAB 7.0 版起，指令窗中的字符、数值等便采用了更为醒目的分类显示：

(1) 对于输入指令中的 if、for、end 等控制数据流的 MATLAB 关键词自动地采用蓝色字体显示；

(2) 对于输入指令中的非控制指令、数码，都自动地采用黑色字体显示；

(3) 输入的字符串自动地采用紫色字体显示。

2. 运算结果的显示

在指令窗中显示的输出有：指令执行后，数值结果采用黑色字体输出；而运行过程中的警告信息和出错信息用红色字体显示。

运行中，屏幕上最常见到的数字输出结果由 5 位有效数字构成。这是“双精度”数据的默认输出格式。用户不要误认为，运算结果的精度只有 5 位有效数字。实际上，MATLAB 的数值数据通常占用 64 位(bit)内存，以 16 位有效数字的“双精度”进行运算和输出。MATLAB 为了比较简洁、紧凑地显示数值输出，才默认地采用“format short g”格式显示出 5 位有效数字。用户根据需要，可以在 MATLAB 指令窗中，直接输入相应的指令，或者在菜单弹出框中进行选择，都可获得所需的数值计算结果显示格式。MATLAB 数值计算结果显示格式的类型见表 1.4－1。

表 1.4-1 数据显示格式的控制指令

指 令	含 义	举例说明
format format short	通常保证小数点后四位有效，最多不超过7位；对于大于1 000的实数，用5位有效数字的科学记数形式显示	314.159 被显示为 314.1590； 3141.59 被显示为 3.1416e+003
format long	小数点后15位数字表示	3.141592653589793
format short e	5位科学记数表示	3.1416e+00
format long e	15位科学记数表示	3.14159265358979e+00
format short g	从 format short 和 format short e 中自动选择最佳记数方式	3.1416
format long g	从 format long 和 format long e 中自动选择最佳记数方式	3.14159265358979
format rat	近似有理数表示	355/113
format hex	十六进制表示	400921fb54442d18
format +	显示大矩阵，正数、负数、零 分别用"+"，"－"和"空格"表示	+
format bank	(金融)元、角、分表示	3.14
format compact	显示变量之间没有空行	
format loose	在显示变量之间有空行	

说明

- format short 显示格式是默认的显示格式。
- 表 1.4-1 中实现的所有格式设置仅在 MATLAB 的当前执行过程中有效。

3. 显示方式的永久设置

用户可以根据需要，对指令窗的字体风格、大小、颜色和数值计算结果显示格式进行设置。设置方法是：选中 {File:Preferences} 下拉菜单项，引出一个参数设置对话框；在此弹出对话框的左栏选中"Font & Colors"，对话框的右边就出现相应的选择内容；用户根据需要和对话框提示对数据显示格式，或字体等进行选择；最后单击 [OK]键完成设置。注意：设置立即生效，并且将被永久保留，即这种设置不因 MATLAB 关闭和开启而改变，除非用户进行重新设置。

在此还要指出，对于数值显示格式的设置，也可以直接在指令窗中，通过指令的运作进行。但这样的设置仅对当前的 MATLAB 指令窗起作用，一旦 MATLAB 关闭，这种设置也就随之失效。

1.4.2 指令行中的标点符号

通过前面算例，读者可能已对标点符号的作用有所体会。在此要强调指出：标点在 MATLAB中的地位极其重要。为此，把各标点的作用归纳成表 1.4-2。

表 1.4-2 MATLAB 常用标点的功能

名 称	标 点	作 用
空 格		（为机器辨认）用作输入量与输入量之间的分隔符； 数组元素分隔符
逗 号	,	用作要显示计算结果的指令与其后指令之间的分隔； 用作输入量与输入量之间的分隔符； 用作数组元素分隔符号
黑 点	.	数值表示中，用作小数点； 用于运算符号前，构成“数组”运算符
分 号	;	用于指令的“结尾”，抑制计算结果的显示； 用作不显示计算结果指令与其后指令的分隔； 用作数组的行间分隔符
冒 号	:	用以生成一维数值数组； 用做单下标援引时，表示全部元素构成的长列； 用做多下标援引时，表示那维上的全部元素
注释号	%	由它“启首”的所有物理行部分被看作非执行的注释
单引号对	' '	字符串记述符
圆括号	( )	改变运算次序； 在数组援引时用； 函数指令输入宗量列表时用
方括号	[ ]	输入数组时用； 函数指令输出宗量列表时用
花括号	{ }	胞元数组记述符； 图形中被控特殊字符括号
下连符	_	可用作一个变量、函数或文件名中的连字符，以提高可读性； 图形中被控下脚标前导符
续行号	...	由三个以上连续黑点构成，把其下的物理行看作该行的“逻辑”继续，以构成一个“较长”的完整指令
“At”号	@	放在函数名前，形成函数句柄； 匿名函数前导符； 放在目录名前，形成“用户对象”类目录

说明

- 为确保指令正确执行，以上符号一定要在英文状态下输入。MATLAB 不能识别含有中文标点的指令。
- 关于它们的更详细的帮助信息，可在 MATLAB 帮助浏览器左侧 Contents 页的 <MATLAB/ User Guide/ Programming Fundamentals/ Basic Program Components/ Symbol Reference>找到。

1.4.3　指令窗的常用控制指令

指令窗的常用指令及其含义列于表1.4-3中。

表1.4-3　常见的通用操作指令

指　令	含　义	指　令	含　义
ans	最新计算结果的默认变量名	edit	打开M文件编辑器
cd	设置当前工作目录	exit	关闭/退出 MATLAB
clf	清除图形窗	help	在指令窗中显示帮助信息
clc	清除指令窗中显示内容	more	使其后的显示内容分页进行
clear	清除 MATLAB 工作空间中保存的变量	quit	关闭/退出 MATLAB
dir	列出指定目录下的文件和子目录清单	return	返回到上层调用程序；结束键盘模式
doc	在 MATLAB 浏览器中，显示帮助信息	type	显示指定M文件的内容
diary	把指令窗输入记录为文件	which	指出其后文件所在的目录

说明

- 表1.4-3所列的是基本的指令，它们对MATLAB各版都适用。
- 尽管随版本的升级，不断增添着列表中指令的"等价"菜单选项操作或工具条图标操作，但这种"等价"仅对"人机交互"过程而言。至于这些指令在M文件中的作用仍是不可替代的。
- cd及dir指令的操作响应，可以用MATLAB操作桌面上或当前目录浏览器中的"浏览键(Browser)"替代。关于当前目录浏览器的使用，可参见第1.6节。
- clear清除内存变量的操作，可以等价地在"工作空间浏览器"交互界面中实现。关于工作空间浏览器的使用，请看第1.7节。
- edit指令的等价操作是：选择MATLAB操作桌面或指令窗的下拉菜单项{File:New:M-file}，或单击相应工具条上的 和 图标。关于M文件编辑器的使用，可参见第1.9节。

1.4.4　指令窗中指令行的编辑

为了操作方便，MATLAB不但允许用户在指令窗中对输入的指令行进行各种编辑和运行，而且允许用户对过去已经输入的指令行进行回调、编辑和重运行。具体的操作方式见表1.4-4。

表 1.4-4 MATLAB 指令窗中实施指令行编辑的常用操作键

键 名	作 用	键 名	作 用
↑	前寻式调回已输入过的指令行	Home	使光标移到当前行的首端
↓	后寻式调回已输入过的指令行	End	使光标移到当前行的尾端
←	在当前行中左移光标	Delete	删去光标右边的字符
→	在当前行中右移光标	Backspace	删去光标左边的字符
PageUp	前寻式翻阅当前窗中的内容	Esc	清除当前行的全部内容
PageDown	后寻式翻阅当前窗中的内容		

说明

- 表 1.4-4 所列的操作对 MATLAB 各版均适用。
- 事实上，MATLAB 把指令窗中输入的所有指令都记录在内存中专门开辟的“指令历史空间(Command History)”中，只要用户对它们不进行专门的删除操作，它们既不会因为用户对指令窗进行“清屏”操作(即运行 clc 指令)而消失，也不会因用户对“工作空间”进行“清除内存变量”(即运行 clear 指令)而消失。
- 指令窗中输入过的所有指令都被显示在“历史指令浏览器”交互界面中，以供随时观察和调用。关于“历史指令浏览器”的使用可参见第 1.5 节。

例【1.4-1】 指令行操作过程示例。

(1) 若用户想计算 $y_1=\dfrac{2\sin(0.3\pi)}{1+\sqrt{5}}$ 的值，那么用户应依次键入以下字符：

```
y1 = 2 * sin(0.3 * pi)/(1 + sqrt(5))
```

(2) 按 [Enter]键，该指令便被执行，并给出以下结果：

```
y1 =
    0.5000
```

(3) 通过反复按键盘的箭头键，实现指令回调和编辑，进行新的计算。

若又想计算 $y_2=\dfrac{2\cos(0.3\pi)}{1+\sqrt{5}}$，用户当然可以像前一个算例那样，通过键盘把相应字符一个一个“敲入”。但也可以较方便地用操作键获得该指令，具体办法是：先用[↑]键调回已输入过的指令 y1=2 * sin(0.3 * pi)/(1+sqrt(5))；然后移动光标，把 y1 改成 y2；把 **sin** 改成**cos**；再按 [Enter]键，就可得到结果。即

```
y2 = 2 * cos(0.3 * pi)/(1 + sqrt(5))
y2 =
    0.3633
```

说明

可以借助“历史指令窗”进行历史指令的再运行，相关内容可参见第 1.5.1 节。

1.5 Command History 历史指令窗

MATLAB所拥有的丰富资源和友善灵活的环境特别适于用来验证一些思想，思考一些问题，并帮助进行创造性思维。用户可以在MATLAB环境中，边想边做，做做想想，对随时蹦出的思想“火花”可即刻通过计算加以验证。历史指令(Command History)窗就是为这种应用方式设计的。

1.5.1 历史指令窗简介

历史指令窗记录着每次开启MATLAB的时间，及开启MATLAB后在指令窗中运行过的所有指令行。该窗不但能清楚地显示指令窗中运行过的所有指令行，而且所有这些被记录的指令行都能被复制，或再运行。关于历史指令窗的功能详见表1.5-1。

表1.5-1 历史指令窗主要应用功能的操作方法

应用功能	操作方法	简捷操作方法
单行或多行指令的复制	点亮单行或多行指令；右击引出现场菜单；选中{Copy}菜单项，即可用复合键[Ctrl + V]把它“粘贴”到任何地方(包括指令窗)	
单行指令的运行	点亮单行指令；右击引出现场菜单；选中{Evaluate Selection}菜单项，即可在指令窗中运行，并见到相应结果	双击单行指令
多行指令的运行	点亮多行指令；右击引出现场菜单；选中{Evaluate Selection}菜单项，即可在指令窗中运行，并见到相应结果(详见例1.5-1)	
把多行指令写成M文件	点亮多行指令；右击引出现场菜单；选中{Create M-File}菜单项，就引出书写着这些指令的M文件编辑调试器；再进行相应操作，即可得所需M文件	

1.5.2 历史指令的再运行

历史指令的重新调用，既可以采用第1.4.4节所介绍的方法实现，也可以借助历史指令窗进行。在许多场合，后者显得更为方便、直观。

例【1.5-1】 演示如何再运行算例1.3-10中的全部绘图指令。

具体操作过程：先利用组合操作[Ctrl+鼠标左键]点亮如图1.5-1所示历史指令窗中的那7行指令；当鼠标光标在点亮区时，右击引出现场菜单；选中现场菜单项{Evaluate Selection}，计算结果就出现在指令窗中。

说明

- 历史指令的复制操作步骤大抵相同，但在现场菜单中，应选{Copy}项。
- 单行历史指令的再运行操作更简单，只要用鼠标双击所需的那行指令即可。

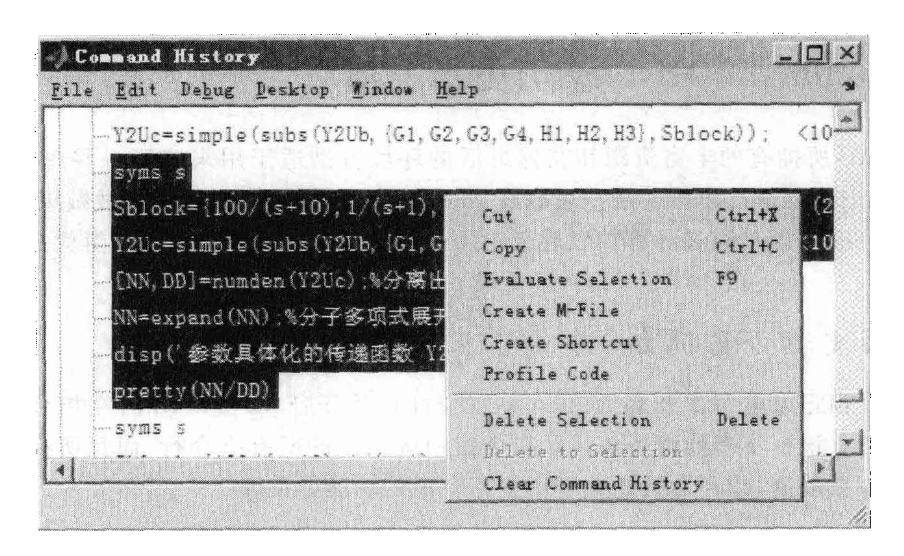


图 1.5-1 再运行历史指令的演示

1.6 Current Directory、路径设置器和文件管理

当在指令窗中运行一条指令时，MATLAB 是怎样从庞大的函数和数据库中，找到所需的函数和数据的呢？用户怎样才能保证自己所创建的文件能得到 MATLAB 的良好管理，又怎样能与 MATLAB 原有环境融为一体呢？这就是本节要介绍的内容。

1.6.1 Current Directory 当前目录浏览器简介

如图 1.6-1 所示的当前目录浏览器界面，自上而下分别是：当前目录名，工具条，文件、文件夹列表及文件描述区等。此外，MATLAB 还为当前目录窗设计了一个专门的操作菜单。借助该菜单可方便地打开或运行 M 文件、装载 MAT 文件数据等，其功能及操作方法详见表 1.6-1。

表 1.6-1 当前目录适配菜单的应用

应用功能	操作方法	简捷操作方法
运行 M 文件	点亮待运行文件；右击引出现场菜单；选中{Run}菜单项，即可使该 M 文件运行	
编辑 M 文件	点亮待运行文件；右击引出现场菜单；选中{Open}菜单项，此 M 文件就出现在编辑/调试器中	双击 M 文件
把 MAT 文件全部数据输入内存	点亮待装数据文件；右击引出现场菜单；选中{Open}菜单项，此文件的数据就全部装入工作内存	双击 MAT 文件
把 MAT 文件部分数据输入内存	点亮待装载数据文件；右击引出现场菜单；选中{Import Data}菜单项，引出数据预览选择对话框“Import Wizard”；在此框中“勾选”待装数据变量名，点击[Finish]键，就完成操作	

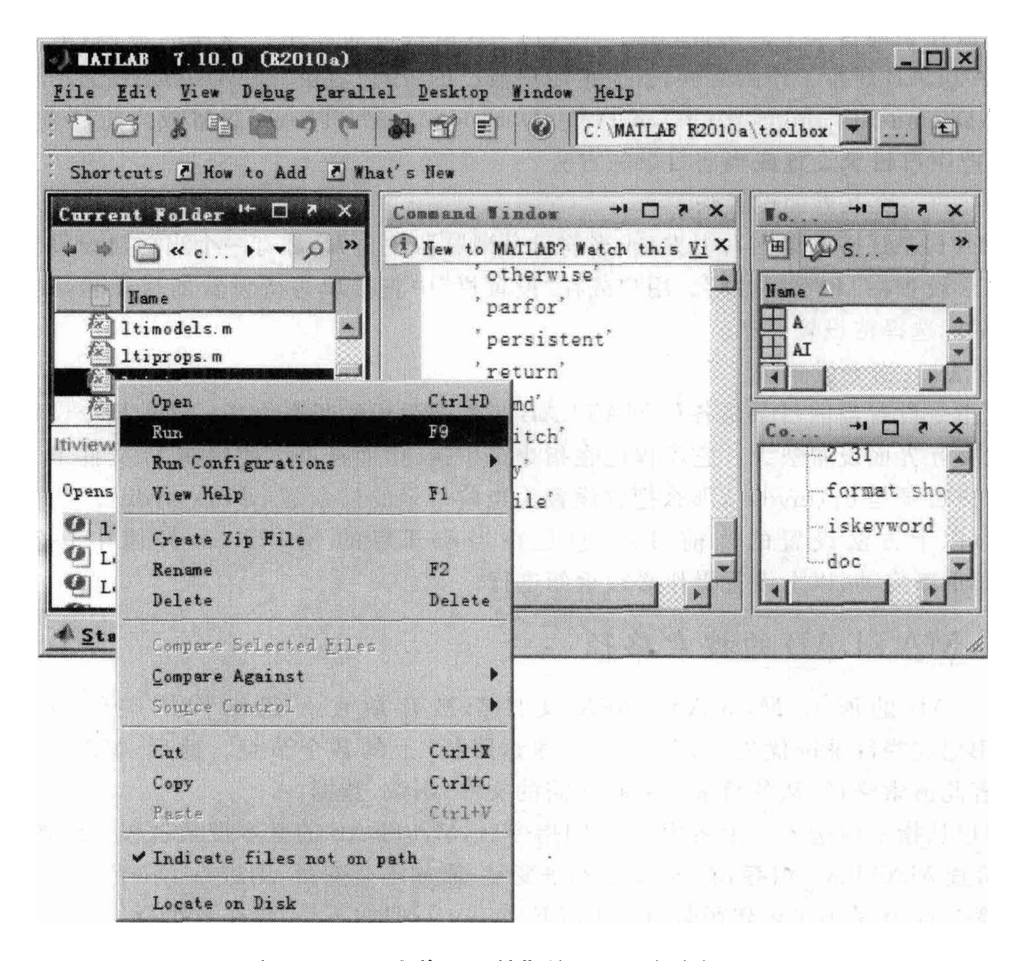


图 1.6-1　当前目录浏览器和适配的弹出菜单

说明

- MATLAB 启动后的默认当前目录通常是：C:\Documents and Setting\user\My Documents\MATLAB。应当指出：在该默认当前目录上存放用户文件是允许的、完全的、可靠的，MathWorks 公司之所以设计这样一个目录，就是供用户使用的。
- 若使用 notebook 文档启动 MATLAB 窗口，则当前目录将是 MATLAB 所在的根目录。提醒读者：千万不要把 MATLAB 所在根目录设成当前目录。对此，用户应该通过重新设置，把当前目录设置在适当的目录上。

1.6.2　用户目录和当前目录设置

(1) 用户目录

MATLAB R2010a 在安装过程中，会自动生成一个目录 C:\Documents and Settings\acer\My Documents\MATLAB。该目录专供用户存放自己的各类 MATLAB 文件。假若用户想另建一个工作目录，采用 Windows 规范操作即可。

(2) 应把用户目录设置成当前目录

在 MATLAB 环境中，如果不特别指明存放数据和文件的目录，那么 MATLAB 总默认地

将它们存放在当前目录上。因此，出于 MATLAB 运行可靠和用户方便的考虑，本书作者建议：在 MATLAB 开始工作的时候，就应把用户自己的“用户目录”或 MATLAB 为用户自动开设的“C:\Documents and Settings\acer\My Documents\MATLAB”设置成当前目录。

(3) 把用户目录设置成当前目录的方法

● 方法一：交互界面设置法。

在 MATLAB 操作桌面右上方，或当前目录浏览器左上方，都有一个当前目录设置区。它包括：“目录设置栏”和“浏览键”。用户或在“设置栏”中直接填写待设置的目录名，或借助“浏览键”和鼠标选择待设置目录。

● 方法二：指令设置法。

通过指令设置当前目录是各种 MATLAB 版本都适用的基本方法。这种指令设置法的适用范围比交互界面设置法大。它不仅能在指令窗中执行，而且可以使用在 M 文件中。假设待设置的用户目录是 c:\mydir，那么把它设置为当前目录的指令是 **cd c:\mydir**。

注意：以上方法设置的当前目录，只是在当前开启的 MATLAB 环境中有效。一旦 MATLAB重新启动，以上设置操作必须重新进行。

1.6.3 MATLAB 的搜索路径

MATLAB 的所有 M，MAT，MEX 文件都被存放在一组结构严整的目录树上。MATLAB把这些目录按优先次序设计为“搜索路径”上的各个节点。此后，MATLAB 工作时，就沿着此搜索路径，从各目录上寻找所需的文件、函数、数据。

当用户从指令窗送入一个名为 cont 的指令后，MATLAB 的基本搜索过程大致如下。

● 检查 MATLAB 内存，看 cont 是不是变量；假如不是变量，则进行下一步。

● 检查 cont 是不是内建函数(Built-in Function)；假如不是，再往下执行。

● 在当前目录上，检查是否有名为 cont 的 M 文件存在；假如不是，再往下执行。

● 在 MATLAB 搜索路径的其他目录中，检查是否有名为 cont 的 M 文件存在。

应当指出：第一，实际搜索过程远比前面描述的基本过程复杂，但有一点可以肯定，凡不在搜索路径上的内容，不可能被搜索；第二，指令 exist，which，load 执行时，也都遵循搜索路径定义的先后次序。

1.6.4 MATLAB 搜索路径的扩展

1. 何时需要修改搜索路径

假如用户有多个目录需要同时与 MATLAB 交换信息，那么就应把这些目录放置在 MATLAB 的搜索路径，使得这些目录上的文件或数据能被调用。又假如其中某个目录需要用来存放运行中产生的文件和数据，那么还应该把这个目录设置为当前目录。

2. 利用设置路径对话框修改搜索路径

采用以下任何一种方法都可以引出设置路径对话框。

● 在指令窗里，运行指令**pathtool**。

● 在 MATLAB 桌面、指令窗等的菜单条中，选择｛File：Set Path｝下拉菜单项，

如图 1.6－2 所示。

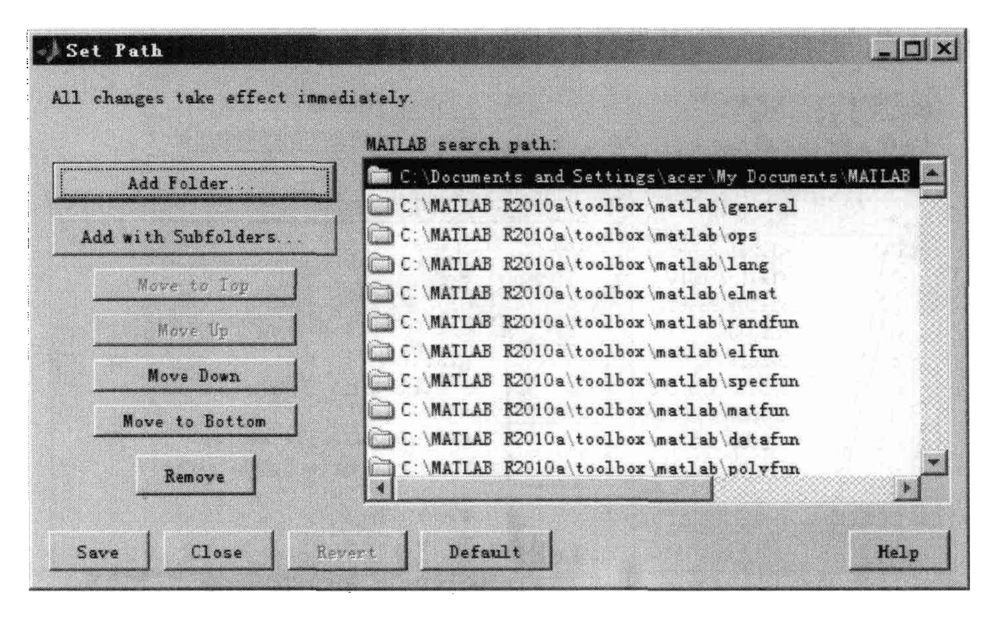


图 1.6－2　路径设置对话框

说明

- 该对话框设置搜索路径有两种修改状态：当前有效修改——假如在路径设置过程中，仅使用了该对话框的左侧按键而不单击[Save]按键；永久有效修改——假如在设置后，单击了对话框下方 [Save]按键。
- 所谓永久有效修改是指：所进行的修改不因 MATLAB 的关闭而消失。

3. 利用指令 path 设置路径

利用 path 指令设置路径的方法对任何版本的 MATLAB 都适用。假设待纳入搜索路径的目录为 c:\my_dir ，那么以下任何一条指令均能实现：

```
path(path,'c:\my dir')         把 c:\my_dir 设置在搜索路径的尾端
path('c:\my dir',path)         把 c:\my_dir 设置在搜索路径的首端
```

说明

- 用 path 指令扩展的搜索路径仅在当前 MATLAB 环境下有效。也就是说：若用户退出当前 MATLAB 后，再重新启动 MATLAB，那么在前一环境下用 path 所定义的扩展搜索路径无效。
- 用 path 指令扩展的搜索路径的方法可以编写在程序中。

1.7　工作空间浏览器和变量编辑器

1.7.1　工作空间浏览器和变量可视化

工作空间浏览器（或称内存浏览器）默认地放置于 MATLAB 操作桌面的左上侧后台。点

击桌面左上侧框下方外露的“Workspace”窗标，可使工作空间浏览器出现在桌面的前台，如图1.7－1所示。该浏览器的功用，详见表1.7－1。

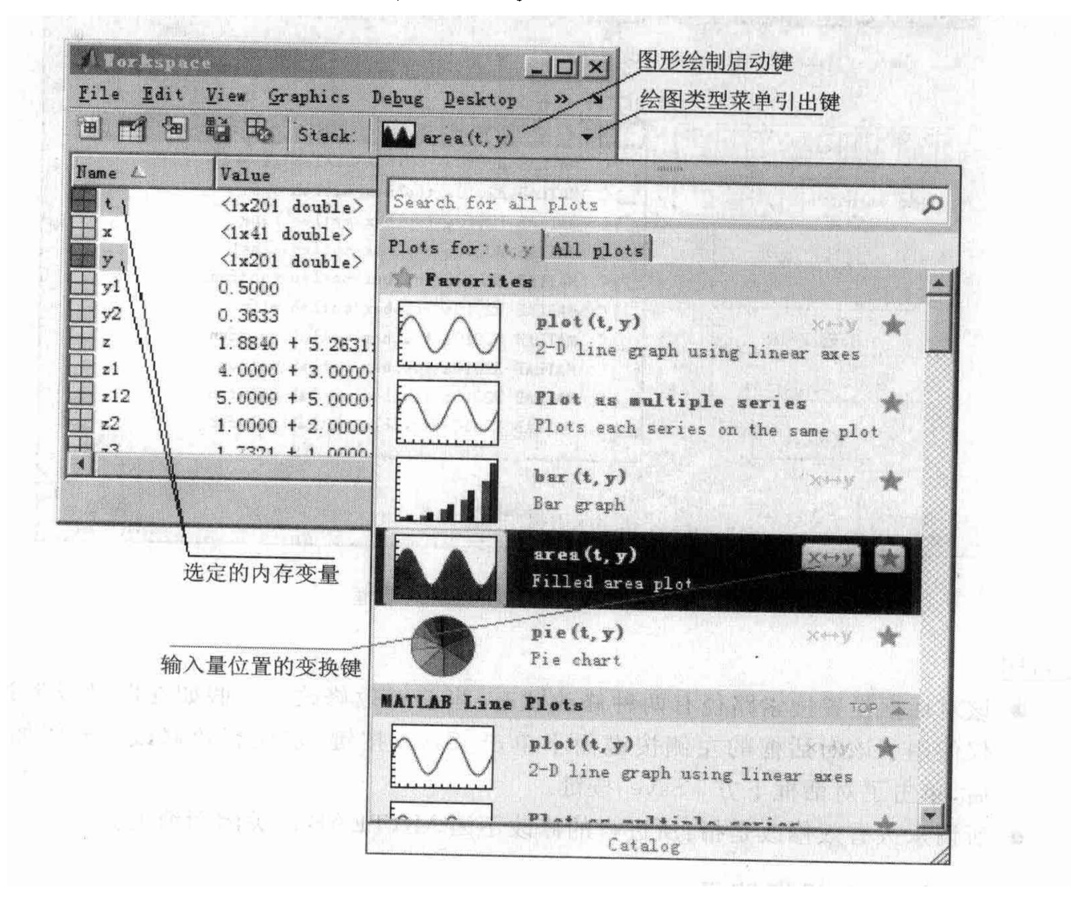


图1.7－1 工作空间浏览器及“绘图工具”图标的展开

【例1.7—1】 通过“工作空间浏览器”的运作，采用图形显示内存变量 t 和 y 之间的关系图形。（注意：本例是在例1.3－10运行后进行的，因此内存中保存着由例1.3－10产生的全部变量。）

（1）绘图变量的选定

在“工作空间浏览器”中，用鼠标点亮所需图示的变量 y 和 t。

（2）选定绘图的类型

点击[绘图类型菜单引出键]，引出绘图类型菜单。假如需要绘制“填色面图”，则用鼠标点中“填色面图”栏（参见图1.7－1）即可。

（3）绘图变量位置的交换

由于在选择变量时，先点选 y，后选 t，所以在绘图指令显示出 area(y,t)。显然，指令 area 的两个输入量位置不正确。为纠正这种错误，用鼠标点击[输入量位置交换键]，参见图1.7－1。

（2）图形的绘制

经过以上操作后，再双击“填色面图”菜单，就绘制出如图1.7－2所示的图形。

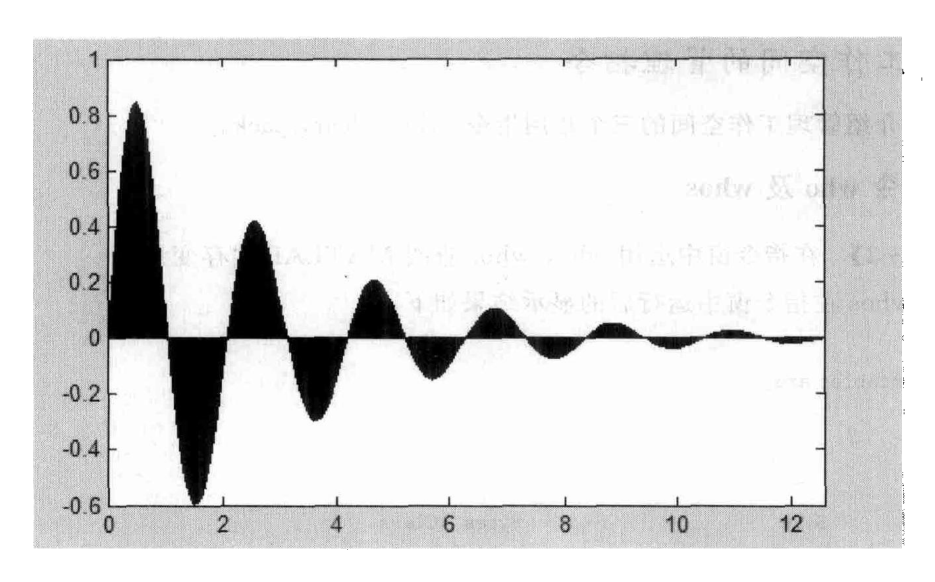


图 1.7-2 数组(t, y)表示的曲线

说明

借助图形表现数据是十分常用的手段。这是由于图形具有表现数据内在联系和宏观特征的卓越能力。正是出于这种考虑,MATLAB 提供了"图示数据"的多种途径。

表 1.7-1 工作空间浏览器主要功能及其操作方法

功 能	操作方法
新变量创建	点击图标,在工作空间中生成一个"unnamed"的新变量;双击该新变量图标,引出 Variable Editor 变量编辑器(参见图 1.7-3);在变量编辑器中,向各元素输入数据;最后,对该变量进行重命名
变量内容显示	点亮变量;或点击图标,或选中弹出菜单中的{Open Selection}项,则变量内含的数据就显示在"Variable Editor"变量编辑器中(参见第 1.7—3 节)
向内存装载文件数据	点击图标;选择 MAT 数据文件;再单击那文件,引出"Import Wizard"界面,它展示文件所包含的变量列表;再从列表中,选择待装载变量,便可
把变量保存进文件	选择待保存到文件的(一个或多个)变量,或点击图标,或选中弹出菜单中的{Save Workspace As}项,便可把那些变量保存到 MAT 数据文件(详见第 1.7-4 节之一)
plot(t,y) 图形绘制启动键	点击 plot(t,y) 键绘制出选定类型的图形(参见例 1.7-1)
绘图类型菜单引出键	点击键,引出绘图类型菜单供选择(参见例 1.7-1)

1.7.2 工作空间的管理指令

本节要介绍管理工作空间的三个常用指令：who，clear，pack。

1. 查询指令 who 及 whos

【例 1.7-2】 在指令窗中运用 who，whos 查阅 MATLAB 内存变量。

who，whos 在指令窗中运行后的显示结果如下：

```
who
Your variables are:

ans  t    y

whos
  Name        Size                    Bytes  Class

  ans         1x1                         8  double
  t           1x201                    1608  double
  y           1x201                    1608  double
```

说明

- who，whos 指令操作对 MATLAB 的所有版本都适用。
- 本例两个指令的差别仅在于获取内存变量信息的简单和详细程度不同。
- 读者运行 who，whos 指令后的变量列表随具体情况而不同。本例的变量列表是在本书作者进行特定操作后产生的。

2. 从工作空间中删除变量和函数的指令 clear

最常用的几种格式：

clear	清除工作空间中的所有变量
clear var1 var2	清除工作空间中的 var1 和 var2 变量
clear all	清除工作空间中所有的变量、全局变量、编译过的 M 函数和 MEX 链接
clear fun1 fun2	清除工作空间中名为 fun1 和 fun2 的函数

注意：在第 2,4 种调用格式中，clear 后面的变量名和函数名之间一定要采用“空格”分隔，而不能采用其他符号。

3. 整理工作空间内存碎片的指令 pack

MATLAB 运行期间，会自动地为产生的变量分配内存，也会为使用到的 M 函数分配内存。有时对于容量较大的变量，会出现“Out of memory”的错误。此时，可能使用 clear 指令清除若干内存中的变量也无济于事。产生这种问题的一个原因是：MATLAB 存放一个变量时，必须使用“连成一片”的内存空间。对于那些被碎片分割得“支离破碎”的内存空间，即便它们的总容量超过待生成变量，也无法使用。在这种情况下，借助 pack 指令也许能解决问题。

1.7.3 Variable Editor 变量编辑器

双击工作空间浏览器中的变量图标，将引出如图 1.7-3 所示的变量编辑器（Variable

Editor)。该编辑器可用来查看、编辑数组元素;对数组中指定的行或列进行图示。

Variable Editor - s

File Edit View Graphics Debug Desktop Window Help

s <3x10 double>

	1	2	3	4	5
1	0.8147	0.9134	0.2785	0.9649	0.9572
2	0.9058	0.6324	0.5469	0.1576	0.4854
3	0.1270	0.0975	0.9575	0.9706	0.8003

图 1.7-3 变量编辑器

单击图标,创建一个名为“unnamed”的变量;在双击该变量引出一个与图 1.7-3 类似的界面。但数组中,除第一个元素为 0 外,其余均为“空白”。利用这个界面,读者就可以比较自在地输入较大的数组。

现在的变量编辑器不但能观察和编辑“双精度”数组,而且也能观察和编辑“字符串”数组、“胞元”数组、和“构架”数组,还能借助“数据链接(Data link)”和“数据刷(Data Brush)”与图形窗中的图形相关联(参见 5.5 节)。

1.7.4 数据文件和变量的存取

1. 借助工作空间浏览器产生保存变量的 MAT 文件

- 从工作空间浏览器中选择待保存到文件的(一个或多个)变量。
- 单击工作空间浏览器工具条图标,或选中弹出菜单中的 {Save As} 项,就弹出 Windows标准的目录和文件名输入对话窗。
- 选定数据文件的保存目录。数据文件应保存在用户选定的目录上,或 MATLAB 自动生成的用户工作目录,即 C:\Documents and Settings\acer\My Documents\MATLAB。注意:假如不有意识地选定目录,那么数据文件将被保存在 MATLAB 的当前目录上。
- 输入数据文件名,如 mydata(注意:.mat 扩展名会自动生成),点击[保存]键,就完成 MAT 数据文件的产生。

2. 借助输入向导 Import Wizard 向工作空间装载变量

- 点击工作空间浏览器上的图标,或 MATLAB Desktop 下拉菜单 {File>Import Data},引出 Windows 标准的目录和文件选择对话窗。
- 在用户的目标目录上,选中 MAT 数据文件(如光盘 For2010a\mfiles 子目录上的 prob_data401.mat);再双击那文件,引出如图 1.7-4 所示的“Import Wizard”界面,它展示出文件所包含的变量列表;再从列表中,通过“勾选”,选择待装载变量(如图中的 t 和 y);再点击[Finish]按键,变量 t 和 y 就被装载到工作空间。

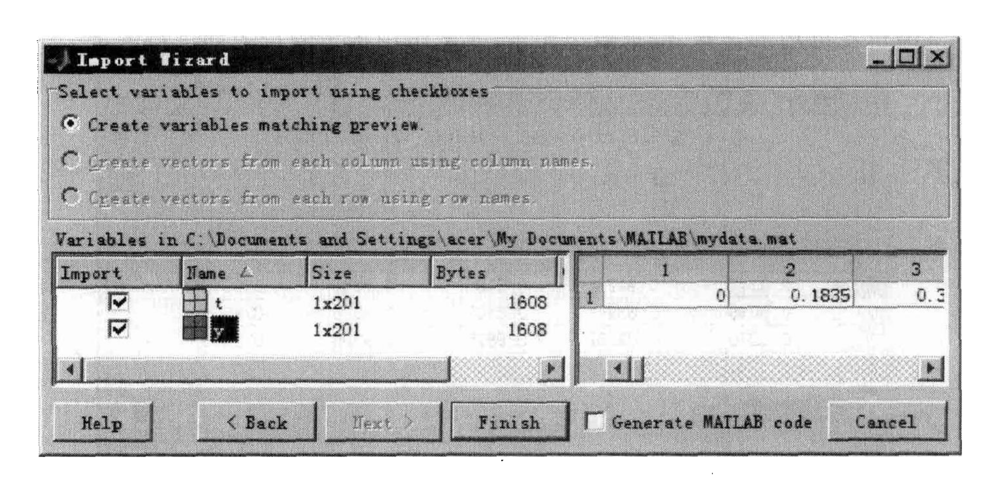


图 1.7-4　向工作空间装载变量的输入向导

3. 存取数据的操作指令 save 和 load

利用 save，load 指令实现数据文件存取是 MATLAB 各版都采用的基本操作方法。它的具体使用格式如下。

save FileName	把全部内存变量保存为 FileName. mat 文件
save FileName v1 v2	把变量 v1，v2 保存为 FileName. mat 文件
save FileName v1 v2 -append	把变量 v1，v2 添加到 FileName. mat 文件中
save FileName v1 v2 -ascii	把变量 v1，v2 保存为 FileName 8 位 ASCII 文件
save FileName v1 v2 -ascii -double	把变量 v1，v2 保存为 FileName 16 位 ASCII 文件
load FileName	把 FileName. mat 文件中的全部变量装入内存
load FileName v1 v2	把 FileName. mat 文件中的 v1，v2 变量装入内存
load FileName v1 v2 -ascii	把 FileName ASCII 文件中的 v1，v2 变量装入内存

说明

- FileName 文件名可以带路径，也可以带扩展名。
- v1 ，v2 代表变量名；指定的变量个数不限，只要内存或文件中存在；变量名与变量名之间必须以空格相分隔。
- -ascii 选项使数据以 ASCII 格式处理。生成的（不带扩展名的）ASCII 文件可以在任何"文字处理器"中被修改。如果数据较多的变量需要进行修改，那么 ASCII 格式的数据文件很适用。
- 如果指令后没有-ascii 选项，那么数据以二进制格式处理。生成的数据文件扩展名一定是 mat。

例【1.7-3】　数据的存取（假定内存中已经存在变量 X,Y,Z）。

（1）建立用户目录，并使之成为当前目录，保存数据；

```
mkdir('c:\','my_dir');         % 在 C 盘上创建目录 my_dir
cd c:\my_dir                   % 使 c:\my_dir 成为当前目录
save   saf X Y Z               % 选择内存中的 X,Y,Z 变量保存为 saf.mat 文件
dir                            % 显示目录上的文件
```

```
.          ..         saf.mat
```

(2) 清空内存，从 saf.mat 向内存装载变量 Z。

```
clear                                  % 清除内存中的全部变量
load  saf Z                            % 把 saf.mat 文件中的 Z 变量装入内存
who                                    % 检查内存中有什么变量
Your variables are:
Z
```

说明

如果一组数据是经过长时间的复杂计算后获得的，那么为避免再次重复计算，常使用 save 加以保存。此后，每当需要，都可通过 load 重新获取这组数据。这种处理模式常在实际中采用。

1.8　Editor/Debugger 和脚本编写初步

对于比较简单的问题或一次性问题，通过指令窗中直接输入一组指令去求解，也许是比较简便、快捷的。但当待解决问题所需的指令较多和所用指令结构较复杂时，或当一组指令通过改变少量参数就可以被反复使用去解决不同问题时，直接在指令窗中输入指令的方法就显得烦琐。M 脚本文件就是设计来解决这个矛盾的。

1.8.1　Editor/Debugger M 文件编辑器简介

默认情况下，如图 1.8－1 所示的 M 文件编辑器（Editor/Debugger）不随 MATLAB 的启动而开启，只有当编写 M 文件时才启动。M 编辑器不仅可以编辑 M 文件，而且可以对 M 文件进行交互式调试；M 文件编辑器不仅可处理带 .m 扩展名的文件，而且可以阅读和编辑其他 ASCII 码文件。

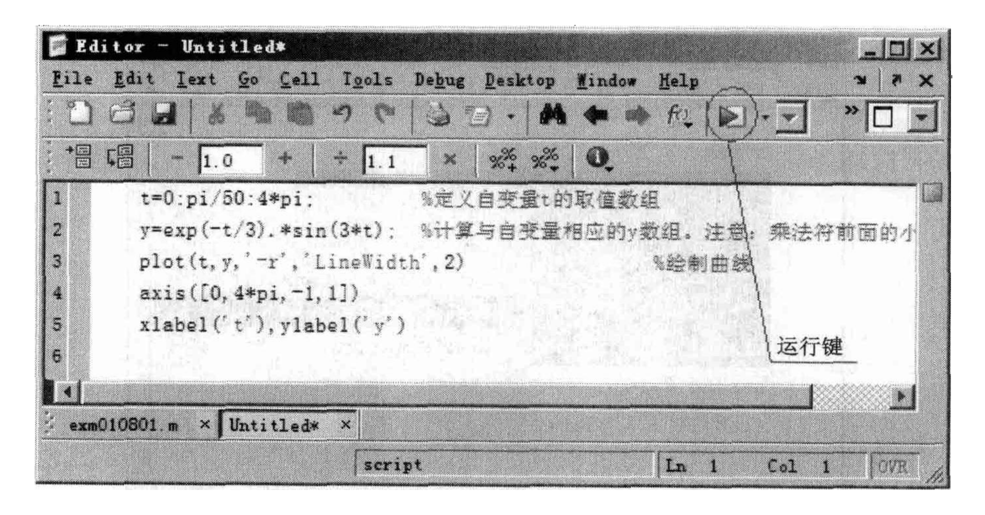


图 1.8－1　M 文件编辑器示图

M 文件编辑器的启动方法有以下几种。

- 单击 MATLAB 桌面上的图标，或选中菜单项 {File>New>M－File}，或直接在指令窗口输入指令 edit，都可以打开空白的 M 文件编辑器。
- 单击 MATLAB 桌面上的图标，或选中菜单项 {File:Open}，可引出“Open”文件选择对话框，在填写所选文件名后，再点击[Open]键，就可引出展示相应文件的 M 文件编辑器。在指令窗中，把待打开文件名写在 edit 后（之间应加一空格），指令运行后，文件编辑器就打开该文件。
- 双击当前目录窗中的所需 M 文件，可直接引出展示相应文件的 M 文件编辑器。

1.8.2 M 脚本文件编写初步

所谓 M 脚本文件是指：第一，该文件中的指令形式和前后位置，与解决同一个问题时在指令窗中输入的那组指令没有任何区别；第二，MATLAB 在运行这个脚本时，只是简单地从文件中读取那一条条指令，送到 MATLAB 中去执行；第三，与在指令窗中直接运行指令一样，脚本文件运行产生的变量都是驻留在 MATLAB 基本工作空间中；第四，文件扩展名是“.m”。

例【1.8－1】 编写解算例 1.3－10 题目的 M 脚本文件，并运行之。

操作步骤：

- 在历史指令窗中，找到算例 1.3－10 的运行指令，并把它们选中点亮，参见图 1.8－2。
- 右击，选中弹出现场菜单中的 {Create M-file}，便引出如图 1.8－1 的 M 文件编辑器。
- 保存文件的操作是 Windows 标准操作。首先，选择 C:\Documents and Settings\acer\My Documents\MATLAB 为文件保存目录，然后以 exm010801 为文件名进行保存，于是就得到了 exm010801.m 文件。
- 因为 C:\Documents and Settings\acer\My Documents\MATLAB 是 MATLAB 安装时自动生成“在搜索路径上”的目录，所以直接单击 M 文件编辑器上的工具图标运行键，就可以得到如图 1.3－4 的曲线。

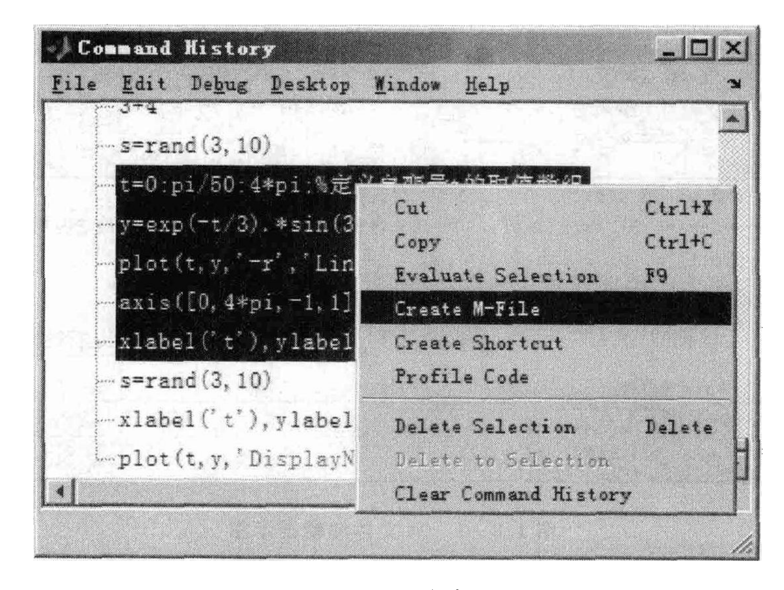


图 1.8－2 利用历史指令创建 M 文件

说明

- 当使用 M 文件编辑调试器保存文件时，或在 MATLAB 指令窗中运行 M 文件时，不必写出文件的扩展名。
- 在 M 文件编辑调试器中，可以采用汉字注释，并总可获得正确显示，参见图 1.8－1。

1.9　帮助系统及其使用

读者接触、学习 MATLAB 的起因、目的不同，借助 MATLAB 所想解决的问题也不同，从而会产生不同的求助需求。对于初学者，最急于知道的是：MATLAB 的基本用法；MATLAB 老用户很想知道的是：MATLAB 新版本有什么新特点、新功能；对科研工作者来说，面对不断变化的实际问题，常常产生两类困惑：知道具体指令但不知道该怎么用，或想解某个具体问题但不知道 MATLAB 有哪些指令可用。

MATLAB 作为一个优秀的科学计算软件，其帮助系统考虑了不同用户的不同需求，构成了一个比较完备的帮助体系。并且，这种帮助体系随 MATLAB 版本的重大升级，其完备性和友善性都会有较大的进步。

作者在此建议：不管以前是否使用过 MATLAB，任何用户都应尽快了解 MATLAB 的帮助系统，掌握各种获取帮助信息的方法。只有这样，用户才可能较好地运用 MATLAB 资源，快捷、可靠、有效地独立解决自己面临的各种问题。

1.9.1　构成帮助体系的三大系统

MATLAB 帮助系统系统及其特点和包含资源列于表 1.9－1 中。

表 1.9－1　MATLAB 的帮助体系

帮助形式	特　点	资　源
指令窗帮助系统	文本形式；最可信、最原始；不适于系统阅读	直接从指令窗中，通过 help 指令获得；所有包含在 M 文件之中的帮助注释内容
帮助导航系统	HTML 形式；系统叙述 MATLAB 规则和用法；适于系统阅读和交叉查阅；最重要的帮助形式	位于 matlab\help 目录下；通过帮助浏览器获得；HTML 和 XML 文件，物理上独立于 M 文件，是次生性帮助文件；本书重点介绍
Web 网帮助系统	包括各种 PDF 文件、视频演示文件、各种讨论组等	mathworks 公司网站；MATLAB 操作界面下拉菜单{Help>Web Resources>}；本书不作详细介绍

1.9.2　常用帮助指令

1. 函数搜索指令

在“知道具体函数指令名称，但不知道该函数如何使用”的情况下，运用函数搜索指令能很好地获得帮助信息。函数搜索指令的调用方法如下：

help	列出所有函数分组名(Topic Name)
help TopicName	列出指定名称函数组中的所有函数
help FunName	给出指定名称函数的使用方法
helpwin	列出所有函数分组名(Topic Name)
helpwin TopicName	列出指定名称函数组中的所有函数
helpwin FunName	给出指定名称函数的使用方法
doc ToolboxName	列出指定名称工具包中的所有函数名
doc FunName	给出指定名称函数的使用方法

说明

- 在此,TopicName, FunName, ToolboxName 分别用来表示待搜索的分组函数名、函数文件名、工具包名。
- help 搜索的资源是 M 文件帮助注释区的内容。这部分资源用纯文本形式写成。它简要地叙述该函数的调用格式和输入输出量含义。该帮助内容最原始,但也最真切可靠。
- helpwin 搜索的资源还是 M 文件帮助注释区的内容。但它的显示形式已不再是"比较简陋的文本",而被自动转换成"比较方便的超文本"。
- doc 搜索是在 HTML 文件构成的帮助子系统中进行的。HTML 文件是根据 M 文件资源编写的,内容比 M 文件帮助注释详细。该子系统,由于采用"超链接"机理,因此检索、查阅比较方便。
- 函数搜索指令的功能与帮助导航器中的"Index 搜索窗"相同。

2. 词条搜索指令

在"想解某具体问题,但不知道有哪些函数指令可以使用"的场合,词条搜索指令也许比较有用。

lookfor KeyWord	对 M 文件 H1 行进行单词条检索
docsearch	对 HTML 子系统进行多词条检索

说明

- lookfor 搜索的资源是 M 文件帮助注释区中的第一行(简称 H1 行)。
- docsearch 指令使用格式
  - ■ 格式:docsearch('Word1 Word2'),或 docsearch('Word1 OR Word2)
    搜索是对"每个词条"按"或"逻辑进行。
  - ■ 格式:docsearch('Word * ')
    * 是通配符,凡词头为 Word 的词条将都被检索。
  - ■ 格式:docsearch(' "Word1 Word2" ')
    将对由 Word1 Woed2 构成的合成词组进行搜索。
- docsearch 搜索是在 HTML 文件构成的帮助子系统中进行的。它的搜索功能强、效率高,搜索到的内容也比较详细。该词条搜索指令的功能与帮助导航器中的"Search 搜

索窗”相同。

1.9.3　Help 帮助浏览器

1. 帮助浏览器的导出

帮助浏览器(Help Brower)搜索的资源是 Mathworks 专门创建的 HTML 随“机”帮助系统。它的内容来源于所有 M 文件,但更详细。它的界面友善,交叉查阅尤其方便,如图 1.9-1 所示。这是用户寻求帮助的最主要资源。

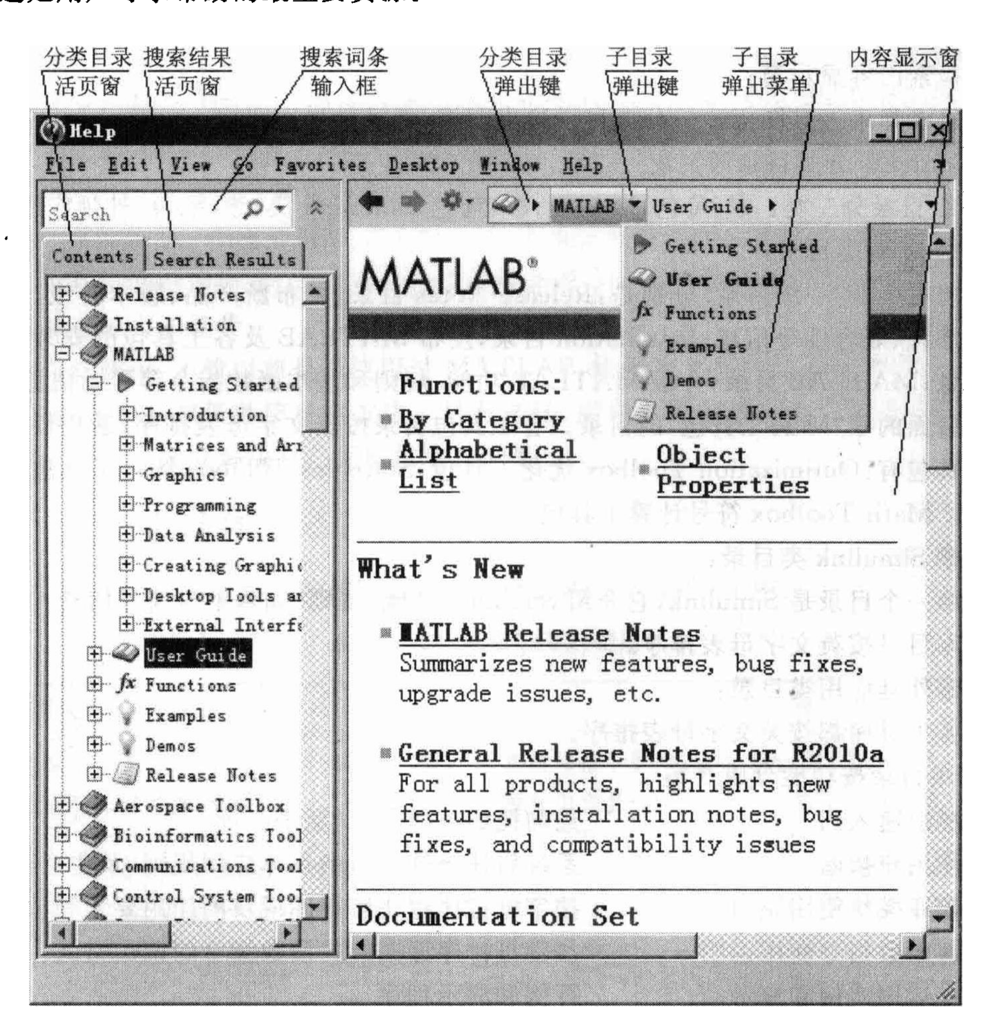


图 1.9-1　默认的帮助浏览器界面

引出帮助浏览器的方法有以下几种：

(1) MATLAB Desktop 操作界面上操作法

方法一：点击工具条的图标；

方法二：选中下拉菜单项 {Help>MATLAB Help>Product Help}。

(2) 指令窗操作法

在指令窗中运行 helpbrowser 或 helpdesk。

2. 帮助浏览器界面简介

(1) 总体外观

图 1.9-1 显示的帮助浏览器界面的结构如下：

- 左侧检索区：
  - Contents 分类目录活页窗；
  - Search Results 检索结果活页窗；
  - (Search)搜索词条输入框。
- 右侧显示区：
  - 检索内容显示窗；
  - 目录弹出工具图标——分类目录弹出图标；子目录弹出图标。

(2) Contents 分类目录活页窗

- 一级目录分三类：(黄色)M 码类；(蓝色)Simulink 类；(绿色)外延应用类。
  - M 码类目录：

    前三个"一级目录"分别是：Release Notes 目录，发布新产品、版本升级、老版本的修订、兼容性等信息；Installation 目录，发布 MATLAB 及各工具包的安装和激活信息；MATLAB 目录，介绍 MATLAB 功能、规则和基本函数指令等帮助信息。

    后面的"M 码工具包"的目录。各工具包目录按英文字母表排序；通用性较强的工具包有：Optimization Toolbox 优化工具包，Statistics Toolbox 统计工具包，Symbolic Math Toolbox 符号计算工具包。
  - Simulink 类目录：

    第一个目录是 Simulink，它介绍 Simulink 功能、规则、和基本模块等信息。其他工具库目录按英文字母表排序。
  - 外延应用类目录：

    各工具库都按英文字母表排序。
- 二级目录按功能分成 5 类
  - 快速入门　　最简捷的入门介绍，新手必读。
  - 用户指南　　系统叙述该软件包的具体应用规则，及注意事项。
  - 库模块使用说明　　按字母排序逐块解释库模块的使用要领和相关连接。
  - 函数指令使用说明　　按字母排序逐条解释函数指令的调用格式。
  - 运用实例和演示　　算例和演示程序。
  - 版本说明　　说明版本新增、更新内容和兼容状况。

(3) 搜索词条输入框

- 在搜索框里，既可以输入函数指令名(如 inv)，又可以输入各种专业词条(如 inverse matrix)。输入确认，按[Enter]键。
- 词条搜索规则
  - 格式：Word1 Word2，或 Word1 OR Word2

    搜索是对"每个词条"按"或"逻辑进行。
  - 格式：Word *

* 是通配符。凡词头为 Word 的词条将都被检索。

■ 格式："Word1　Word2"

将对由 Word1 Word2 构成的合成词组进行搜索

(4) Search Results 搜索活页窗

假如在搜索框中，输入词组 Laplace transform，按[Enter]键进行搜索，那么帮助浏览器将呈现如图 1.9-2 的界面。图中，因为输入词组中的单词被空格分开，所以各单词分别被搜索，并被彩化。

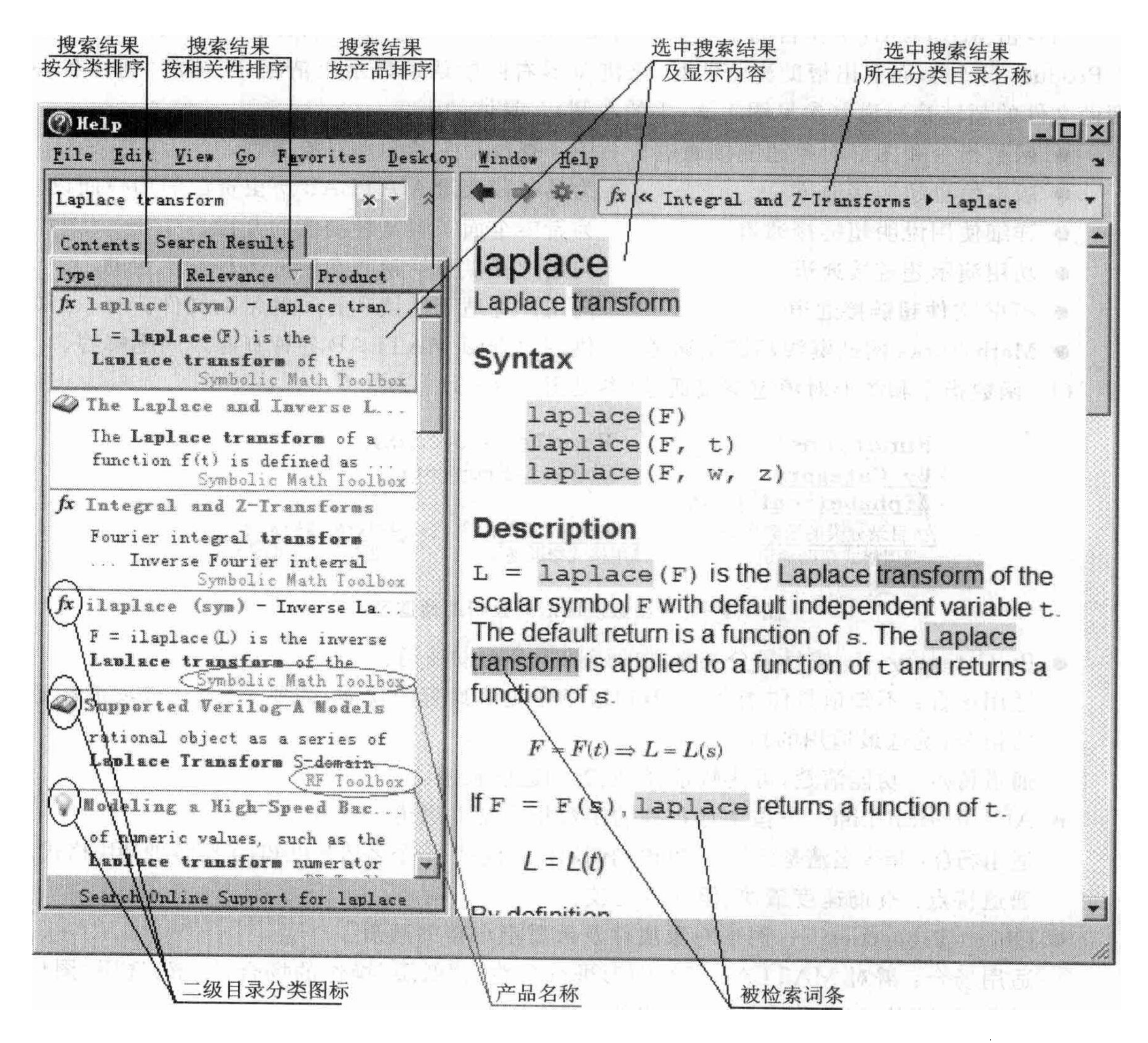


图 1.9-2　按相关性罗列的搜索结果

左侧搜索结果列表有如下三种可能的排列方式：

● Relevance 相关性排列方式

这是默认排列方式，是按照该段文字与“被搜索词组”相关数量大小排列的。这样便于用户找到最集中、详细的帮助信息。

● Type 二级目录类型排列方式

如果用鼠标点击帮助浏览器左侧的[Type]按键，那么搜索结果将按“二级目录类型(参见图 1.9-2)”排列。这样便于用户找到最容易入门的帮助信息。

● Product 产品类型排列方式

如果用鼠标点击帮助浏览器左侧的[Product]按键，那么搜索结果将按产品名称，即工具包名称，字母顺序排列。这样便于用户找到希望工具包中的针对性较强的帮助信息。

3. 帮助浏览器默认显示的利用

当点击 MATLAB 工作台的工具条图标，或选中下拉菜单项 {Help>MATLAB Help>Product Help}，而引出帮助浏览器时，该浏览器右侧默认地显示出清晰、简捷的“通往各种帮助文件的超链接通道”(参见图 1.9-1 的右侧)。具体如下：

● 函数指令和图形对象超链接通道　便于查找指令和图形对象属性。
● 版本信息超链接通道　为具有较大量 MATLAB 历史资源的用户而设。
● 详细使用说明超链接通道　为希望全面了解某种功能而开设。
● 功用演示超链接通道　向用户提供包括视频在内的多种演示帮助。
● PDF 文件超链接通道　向用户更适于阅读和打印的帮助文件。
● MathWorks 网站资源超链接通道　供用户查阅 MATLAB 各种资源或问题解答。

(1) 函数指令和图形对象超链接通道(参见图 1.9-3)

Functions:　　Handle Graphics:
■ By Category　　■ Object Properties
■ Alphabetical List
按目录分类的函数指令调用格式帮助通道　按字母排列的函数指令调用格式帮助通道　图形对象属性及设置指令使用方法帮助通道

图 1.9-3　函数及图形对象帮助通道区

● By Category——按目录分类排列的函数指令帮助通道。

适用场合：不知道具体指令名，但知道指令应该具有的功能；需要了解、比较功能相近的指令；挑选最适用的指令。

通道特点：功能清楚，可比较选择；但查询速度较慢。

● Alphabetical List——按字母排列的函数指令帮助通道。

适用场合：指令名清楚已知，但准确的调用格式模糊；指令名清楚已知，了解各种调用格式。

通道特点：查询速度最快；但缺少比较。

● Object Properties——图形对象属性及舌质指令帮助通道。

适用场合：需对 MATLAB 所绘图形进行个性化“低层”操作的场合；了解、选用“图形对象属性”及“操作指令”调用格式的场合。

通道特点：层次清楚；属性分列清晰；超链接交互查阅方便。

(2) 版本信息超链接通道(参见图 1.9-4)

● MATLAB Release Notes——历史版本变化汇总。

适用场合：MATLAB 新手不必看；需要了解 MATLAB 历史演进；较多、较重要的 MATLAB 历史版本编写的文件资源启用前；MATLAB 历史版本写成文件运行产生不明原因错误的场合。

What's New

历史版本变化汇总　本版的升级变化说明

- **MATLAB Release Notes**
  Summarizes new features, bug fixes, upgrade issues, etc.
- **General Release Notes for R2010a**
  For all products, highlights new features, installation notes, bug fixes, and compatibility issues

图 1.9-4　版本信息帮助通道区

通道特点：对各版变化、补丁修正描述清晰。

- General Release Notes for R2010a——本版本升级变化说明。

  适用场合：MATLAB新手可浏览新版本对环境的要求；利用MATLAB设计较大型软件库的用户；MATLAB历史版本写成文件运行产生不明原因错误的场合。

(3) 详细使用说明超链接通道(参见图1.9-5)

Documentation Set

- **Getting Started** —— 快速入门
- **User Guides** —— 用户指南
- **Getting Help** —— 帮助指南
  Provides instructions for using help functions, the Help browser, and other resources
- **Examples in Documentation** —— 分功能详解
  Lists major examples in the MATLAB documentation
- **Programming Tips** —— 编程技巧
  Provides helpful techniques and shortcuts for programming in MATLAB

图 1.9-5　详细使用说明通道区

- Getting Started——快速入门。

  适用场合：新手应先读和必读，克服生疏感；感受新版本工作环境。

  通道特点：所包含的网络接口通道，可直接观看Mathworks制作的入门视频；所涉材料浅显易懂。

- User Guider——用户指南。

  适用场合：用户指南是所有帮助内容中最重要的部分；建议每个用户快速浏览阅读，以形成对MATLAB编程环境的宏观了解；遇问题时重点研读，以深入理解使程序真实反映数学模型的注意事项。

  通道特点：是数学模型和程序模型之间的桥梁。

- Getting Help——帮助指南。

  适用场合：新手应先读和必读；熟悉旧版者应浏览。掌握和了解MATLAB的各种帮助方式、适用场合。对新手而言，只要能解决所面临的问题，掌握一两种便可。

通道特点：归纳性强，解释简明。

- Examples in Documentation——分功能详解

  适用场合：建议每个用户浏览比较细的分功能列表；学习、掌握某功能，如怎样合并矩阵、怎样消除数据中的确定性趋势等。

  通道特点：既包含基本编程技能，又包含最实用的数学处理方法；每种功能单独分列，适于片段学习。

- Programming Tips——编程技巧

  适用场合：建议用户快速浏览编程技巧；适于快速了解 MATLAB 功能和使用要领。

  通道特点：包含的各条目短小精干。

(4) 功用演示超链接通道(参见图 1.9-6)

Product Demos

MATLAB Demos —— 功能演示
Presents a collection of demos that you can run from the Help browser to help you learn the product

图 1.9-6 功能演示通道区

- MATLAB Demos——功能演示

  适用场合：感受 MATLAB 界面各图标、菜单的功能，以及交互操作手法；了解、学习典型的 M 码文件的指令运用和编写技巧；了解、学习 GUI 图形用户接口的功能和编写技巧。

  通道特点：Video 视频演示需网络支持；GUI 资源及部分 M 码资源所提供的帮助具有独特性；资源既包括"入门引导"型(初学者适宜)，又有"学科专业"型(科研人员适宜)，还有"编程技巧"型(对 MATLAB 较熟悉者适宜)。

(5) PDF 文件超链接通道(参见图 1.9-7)

Printable (PDF) Documentation on the Web

Printable versions of the MATLAB documentation and related papers on the Web —— PDF格式帮助文件

图 1.9-7 PDF 文档通道区

- Printable versions——PDF 格式帮助文件

  适用场合：需要系统阅读、学习 User Guider 用户指南。

  通道特点：需要得到网络支持；该文件编框架和内容与超文本形式的"User Guider 用户指南"相似。

(6) MathWorks 网站资源超链接通道(参见图 1.9-8)

适用场合：了解 MATLAB 产品、第三方相关产品信息；向 MathWorks 公司进行技术咨询。

通道特点：需网络支持。

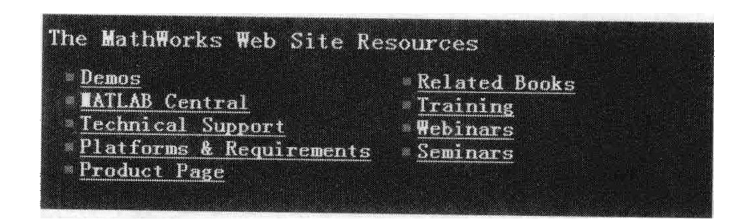


图 1.9-8　MathWorks 网站资源通道区

习题 1

1. 数字 1.5e2，1.5e3 中的哪个与 1500 相同？
2. 请指出如下 5 个变量名中，哪些是合法的？

   abcd－2　xyz_3　3chan　a 变量　ABCDefgh
3. 在 MATLAB 环境中，比 1 大的最小数是多少？
4. 设 a＝－8，运行以下三条指令，问运行结果相同吗？为什么？

```
w1 = a^(2/3)
w2 = (a^2)^(1/3)
w3 = (a^(1/3))^2
```

5. 指令 clear，clf，clc 各有什么用处？
6. 以下两种说法对吗？(1)“MATLAB 的数值表达精度与其指令窗中的数据显示精度相同。”(2)“MATLAB 指令窗中显示的数值有效位数不超过 7 位。”
7. 想要在 MATLAB 中产生二维数组 $\boldsymbol{S}=\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$，下面哪些指令能实现目的？

```
S=[1,2,3;4,5,6;7,8,9]
S=[1  2  3;4  5  6;7  8  9]
S=[1,2,3;4,5,6;7,8,9]     % 整个指令在中文状态下输入
```

8. 试为例 1.3-5 编写一个解题用的 M 脚本文件。

第2章 符号计算

所谓符号计算是指:解算数学表达式、方程时,不是在离散化的数值点上进行,而是凭借一系列恒等式和数学定理,通过推理和演绎,获得解析结果。这种计算建立在数值完全准确表达和推演严格解析的基础之上,因此所得结果是完全准确的。

本书之所以把符号计算内容放在第2章,是出于以下考虑:一,相对于MATLAB的数值计算“引擎”和“函数库”而言,符号计算的“引擎”和“函数库”是独立的;二,在相当一些场合,符号计算解算问题的指令和过程,显得比数值计算更自然、更简明;三,大多数理工科学生在学过高等数学和其他专业基础课以后,比较习惯符号计算的解题理念和模式。

在编写本章时,作者在充分考虑符号计算独立性的同时,还考虑了章节的独立性、完整性。为此,本章不但全面地阐述符号计算,而且在最后一节还详细叙述了符号计算结果的可视化。这样的安排,将使读者在阅读完本章后,就可以运用MATLAB的符号计算能力去解决一些具体问题。

2.1 符号对象和符号表达式

从经典教科书中可知,数学表达式和方程的基本组成是:数字、参数、变量;运算符号(加减乘除等);数学函数(如三角函数、指数函数等)。MATLAB作为面向对象的科学计算语言,也是依靠基本符号对象(包括数字、参数、变量)、运算符及一些预定义函数来构造和衍生符号表达式、符号方程的。

2.1.1 符号对象的创建和衍生

1. 生成符号对象的基本规则

MATLAB规则:

- 任何基本符号对象(数字、参数、变量、表达式)都必须借助专门的符号函数指令sym或syms定义。
- 任何包含符号对象的表达式或方程,将继承符号对象的属性。换句话说,任何包含符号对象的表达式、方程也一定是符号对象。

2. 符号数字

众所周知,一般有限字长数字计算机在记述数字时,不能始终保证完全准确。但在符号计

算中,需要参与计算的数字完全准确。为了区别于数字的一般有限字长记录,MATLAB 符号计算时采用的数字是所谓的符号(类)数字。它的定义格式为:

sym('Num')　　创建一个符号数字 Num(在符号表达式中使用)

sc=sym('Num')　　创建一个符号常数 sc,该常数值准确等于 Num

说明

- 在此,Num 代表一个具体的数字。
- Num 必须处于(英文状态下的)单引号内,构成字符串(关于字符串参见附录 A.1)。

例【2.1-1】 符号(类)数字与数值(类)数字之间的差异。本例演示:它们在创建方式、显示形式、类别判断和具体数值上的不同。

```
a = pi + sqrt(5)                    % 创建一个数值类常数
sa = sym('pi + sqrt(5)')            % 创建一个符号类常数
Ca = class(a)                       % 判断 a 的数据类别
Csa = class(sa)                     % 判别 sa 的数据类别
vpa(sa - a)                         % 在 32 位精度意义上计算两类数字之间的差
a =
    5.3777
sa =
pi + sqrt(5)
Ca =
double
Csa =
sym
ans =
0.0000000000000000138223758410852000485935425641 88
```

说明

- 值得指出:由于 MATLAB 对数值类数字的存储和运算非常讲究,因此在许多情况下,两类数字是完全相同的。
- 本例表现的是:符号类数字总是被准确记录和运算的;而数值类数字并不总能保证被完全准确存储,运算时也会引进截断误差。

3. 基本符号变量

在经典教科书里,常把表达式 $e^{-ax}\sin bx$ 中的 a,b 称谓参数,而把 x 称作变量。在 MATLAB 的符号计算中,a,b,x 都统称为"基本符号变量"。而当对符号表达式进行求解、绘图等操作时,假如不做专门设定,那末 x 总被默认为"待解符号变量"或称"自由符号变量",而其他的基本符号变量被作为"符号参数"处理。

下面介绍几种定义基本符号变量的指令格式:

para = sym('para')　　定义单个复数域符号变量 para

para = sym('para', 'Flag')　　定义单个 Flag 指定域符号变量 para

syms para　　定义单个复数域符号变量 para(的另一种方式)

syms para Flag	定义单个 Flag 指定域符号变量 para(的另一种方式)
syms para1 para2 paraN	定义多个复数域符号变量 para1，para2，paraN
syms para1 para2 paraN Flag	定义多个 Flag 指定域符号变量 para1，para2，paraN

说明

- para，para1，para2，paraN 分别代表(基本)符号变量名。在没有 Flag 指定具体域名的情况下，MATLAB 把它们默认为"复数域符号变量"。
- Flag 代表数域的限定性假设。它可具体取以下关键词：real 表示"实数域"；positive 表示"正实数域"。关于变量限定性假设的更详细描述，请参见第 2.1.5 节。
- 在利用 syms 定义多个(基本)符号变量时，各变量名和变量名之间、变量名和数域限定词之间只能用"空格"分隔。

4. 自由符号变量

基本符号变量可分为：自由符号变量和符号参数。解题通常是围绕自由符号变量进行的，而解得的结果通常是"用符号参数构成的表达式表述自由符号变量"。

解题时，自由符号变量可以"人为指定"，也可以经由软件"默认地自动认定"。本节将通过介绍 symvar 指令的功能，帮助读者理解软件自动认定的默认规则：在没有专门指定变量名的符号运算中，MATLAB 将按照与小写字母 x 的 ASCII 码距离自动识别自由符号变量。此后的解题将围绕那被自动识别的变量进行。

symvar(expression)	列出表达式中的所有基本符号变量
symvar(expression, n)	列出表达式中认定 n 个自由符号变量

说明

- expression 可以是符号表达式，或符号表达式矩阵。此时，该指令对自由变量的确认是对整个矩阵进行的，而不是对矩阵元素逐个进行的。这个性质将在本章以下几节中得到表现。
- 在 MATLAB 符号计算中，x 是首选自由符号变量，其后的次序排列规则是：与 x 的 ASCII 码值之差的绝对值小的字母优先；差绝对值相同时，ASCII 码值大的字母优先。
- symvar 识别自由符号变量时，字母的优先次序为 x，y，w，z，v 等。

例【2.1-2】　用符号计算研究方程 $\sin(3)uz^2+vz+3w-a_5=0$ 的解。本例演示：符号常数、基本符号变量、自由符号变量的区别；symvar 两种调用格式的不同功能；MATLAB 符号计算默认识别自由符号变量的能力；同一方程，指定的待解变量不同，所得解也不同。

(1) 产生符号表达式

```
syms u v w z a5                         % 定义符号参数和变量
f = sym('3');                           % 定义符号常数
Eq = sin(f) * u * z^2 + v * z + f * w - a5;  % 构成符号表达式
```

(2) 基本符号变量和自由符号变量的认定

```
symvar(Eq)                              % 按字母表次序列出基本符号变量，注意没有 f
ans =
[ a5, u, v, w, z]
```

```
symvar(Eq,100)                      % 按离 x 的距离列出所有自由符号变量。
                                    % 为此,作者为方便,把第二个输入量取成 100。
                                    % 实际上,第二输入量只要不小于"真实自由变量数"即可

ans =
[ w, z, v, u, a5]

symvar(Eq,1)                        % 指定识别一个自由符号变量
ans =
w
```

(3) 关于自由符号变量解方程

```
result_1 = solve(Eq)                % solve(Eq)意味着                              <7>
                                    % 关于 w 解方程 sin(f) * u * z^2 + v * z + f * w - a5 = 0
result_1 =
a5/3 - (v * z)/3 - (u * sin(3) * z^2)/3
```

(4) 把 z 指定为待解变量

```
result_2 = solve(Eq,z)      % 对于指定自由符号变量 z 解方程
result_2 =
 -(v - (v^2 + 4 * a5 * u * sin(3) - 12 * u * w * sin(3))^(1/2))/(2 * u * sin(3))
 -(v + (v^2 + 4 * a5 * u * sin(3) - 12 * u * w * sin(3))^(1/2))/(2 * u * sin(3))
```

说明

注意:在本例第〈7〉条指令中,自动识别出的自由符号变量是 w,而不是人们在教科书中比较习惯的 z。

例【2.1-3】 基本符号变量、符号常数、元符号表达式、衍生符号表达式的定义,基本符号变量、自由符号变量的机器辨认。本例目的:体验定义基本符号变量的指令格式,体验衍生符号表达式的创建;感受运行结果的显示形式;符号表达式中检出符号变量的指令格式,和机器所检出的符号变量的排列次序;符号表达式中自由符号变量的机器认定。

(1) 各种符号对象的创建

```
syms a b x X Y                      % 定义基本符号变量
k = sym('3');                       % 定义符号常数
z = sym('c * sqrt(d) + y * sin(t)'); % 创建"元"符号表达式
EXPR = a * z * X + (b * x^2 + k) * Y; % 构成"衍生"符号表达式
```

(2) 列出 EXPR 中的全部基本符号变量

```
symvar(EXPR)                        % 列出所有基本符号变量,k,z 除外
ans =
[ X, Y, a, b, c, d, t, x, y]
```

(3) 列出 EXPR 中全部自由符号变量

```
symvar(EXPR,10)
ans =
[ x, y, t, d, c, b, a, X, Y]
```

(4) 自认定 EXPR 中最优先的一个自由符号变量

```
symvar(EXPR,1)
```

```
ans =
x
```

(5) 自认定 EXPR 中最优先的三个自由符号变量

```
symvar(EXPR,3)
ans =
[ x, y, t]
```

(6) 在利用 sym 创建元符号表达式时应回避 MATLAB 自身的关键词

```
E3 = sym('a * sqrt(theta)')          % theta 作为 sqrt 函数唯一输入量引起问题
??? Error using ==> sym.sym>sym.sym/scalarsym at 382
Error: argument must be of 'Type::Arithmetical' [sqrt]

Error in ==> sym.sym>sym.sym/char2sym at 337
                    Scell = scalarsym(x);

Error in ==> sym.sym>sym.sym/symchar at 182
                Scell = char2sym(x);

Error in ==> sym.sym>sym.sym at 120
                S = cell2sym(S,symchar(x,a,nargin));

E4 = sym('a * sqrt(theta123)')          % 尽可能回避关键词
E4 =
a * theta123^(1/2)

E5 = sym('a * sqrt(theta * t)')          % sqrt 函数中不仅仅包含 theta 时,运行正确
E5 =
a * (t * theta)^(1/2)
```

说明

- 注意:k 是符号常数,因此不是"基本符号变量";z 是"元符号表达式"的,也不是基本符号变量。
- 注意:在利用 sym 创建符号表达式时,注意尽可能避免把"MATLAB 的自用关键词"用作各种符号函数的变量,以免引发"莫名其妙"的错误。

例【2.1-4】 symvar 确定自由变量是对整个矩阵进行的。本例演示:符号矩阵的一种创建方法;矩阵在 MATLAB 中是作为一个整体看待的。

```
syms a b t u v x y
A = [a + b * x,sin(t) + u;x * exp( - t),log(y) + v]          % 创建衍生符号矩阵
symvar(A,1)                                                  % 确定矩阵 A 中的自由符号变量
A =
[  a + b*x, u + sin(t)]
[ x/exp(t), v + log(y)]
ans =
x
```

2.1.2　符号计算中的算符

由于 MATLAB 采用了重载(Overload)技术，使得用来构成符号计算表达式的算符，无论在形状、名称上，还是在使用方法上，都与数值计算中的算符完全相同。下面就符号计算中的基本算符作简要归纳。

(1) 基本运算符

算符“+”，“-”，“*”，“\”，“/”，“^”分别表示“符合矩阵运算法则”的加、减、乘、左除、右除、求幂运算。

算符“.*”，“.\”(或“./”)，“.^”分别表示“数组对应元素间”的乘、除、求幂。

算符“'”，“.'”分别实现矩阵的共轭转置、非共轭转置。

(2) 关系运算符

在符号对象的比较中，没有“大于”、“大于等于”、“小于”、“小于等于”的概念，而只有是否“等于”的概念。

算符“==”，“~=”分别对算符两边的对象进行“相等”、“不等”的比较。当事实为“真”时，比较结果用 1 表示；当事实为“假”时，比较结果则用 0 表示。

2.1.3　符号计算中的函数指令

MATLAB 提供的是面向对象的软件环境。对于不同的数据对象(如数值类和符号类)，它借助重载技术，把具有相同函数计算功能的文件采用同一个函数名加以保存。这样处理后，虽然不同类型数据的处理方法不同，但从形式上看，用于数值计算的函数与用于符号计算的函数却没有什么区别。至于运算中是调用数值计算文件还是符号计算文件，完全由所计算的对象属性(数值类还是符号类)决定。

MATLAB 用于符号计算的函数很多(见表 2.1-1)，大致分为三个层次：

- 第一层次：几乎与所有数值类函数和指令对应的“同名符号类函数和指令”。
- 第二层次：约 50 个经典特殊函数(如误差函数、贝塞尔函数、椭圆积分等)。它们要借助 mfun 调用。在 MATLAB 的数值计算中没有对应的函数可供调用。
- 第三层次：数量很大的 MuPAD 库函数。它们借助 evalin 和 feval 指令调用。

表 2.1-1　MATLAB 中可调用的符号计算函数指令

类　别	情况描述	与数值计算对应关系
基本函数	三角函数、双曲函数及反函数；除 atan2 外	名称和使用方法相同
	指数、对数函数(如 exp, expm)	名称和使用方法相同
	复数函数(注意：没有幅角函数 angle)	名称和使用方法相同
	矩阵分解函数(如 eig, svd)	名称和使用方法相同
	方程求解函数 solve	不同
	微积分函数(如 diff, int)	不完全相同
	积分变换和反变换函数(如 laplace, ilaplace)	只有离散 Fourier 变换
	绘图函数(如 ezplot, ezsurf)	数值绘图指令更丰富

续表 2.1

类　别	情况描述	与数值计算对应关系
经典特殊函数	如误差函数 erf、贝塞尔函数 besselj、第一类完全椭圆积分 EllipticK 等；通过 mfunlist 可以看到所有经典函数名	无对应函数
MuPAD 库函数	借助 evalin 和 feval 指令可调用比 mfunlist 所列范围更广泛的 MuPAD 库函数；需要具备 MuPAD 语言知识	无对应函数

说明

虽然数值计算与符号计算中有许多同名函数，但是读者在使用函数时，还是要十分注意函数对数据类型的要求，否则容易出错。举例来说，就数字而言，有双精度数字和符号类数字之分。这两种数字，显示形式有时非常相似。但假如把符号类数字输入到某个只对数值数据适用的函数（如 plot）中，就一定会产生错误。

2.1.4 符号对象的识别

在 MATLAB 中，函数指令很多。有的函数指令适用于多种数据对象（如数值、符号等），但也有的函数指令只对某种数据对象适用。在数值计算和符号计算混合使用的情况下，由于函数指令与数据对象不适配引起的错误容易发生。为了避免这种错误，MATLAB 提供了用于识别数据对象属性的指令：

class(var)　　给出变量 var 的数据类别（如 double, sym 等）

isa(var,'Obj')　　若变量 var 是 Obj 代表的类别，给出 1，表示“真”

whos　　给出所有 MATLAB 内存变量的属性

例【2.1-5】 数据对象及其识别指令的使用。本例演示：不同数据对象在创建、显示形式、大小性质、内存占用上的差异；演示 class, isa, whos 的具体使用。

（1）生成三种不同类型的矩阵，给出不同的显示形式

```
clear
a=1;b=2;c=3;d=4;                % 产生 4 个数值变量
Mn=[a,b;c,d]                    % 利用已赋值变量构成数值矩阵
Mc='[a,b;c,d]'                  % 字符串中的 a,b,c,d 与前面输入的数值变量无关
Ms=sym(Mc)                      % Ms 是一个符号矩阵,它与前面各变量无关
Mn =
     1     2
     3     4
Mc =
[a,b;c,d]
Ms =
[ a, b]
[ c, d]
```

（2）三种矩阵的大小不同

```
SizeMn=size(Mn)
SizeMc=size(Mc)
```

```
SizeMs = size(Ms)
SizeMn =
     2     2
SizeMc =
     1     9
SizeMs =
     2     2
```

（3）用 class 获得每种矩阵的类别

```
CMn = class(Mn)
CMc = class(Mc)
CMs = class(Ms)
CMn =
double
CMc =
char
CMs =
sym
```

（4）用 isa 判断每种矩阵的类别（若返回 1，表示判断正确）

```
isa(Mn,'double')
isa(Mc,'char')
isa(Ms,'sym')
ans =
     1
ans =
     1
ans =
     1
```

（5）利用 whos 观察内存变量的类别和其他属性

```
whos Mn Mc Ms                    % 观察三个变量的类别和属性
  Name      Size                 Bytes  Class            Attributes

  Mc        1x9                     18  char
  Mn        2x2                     32  double
  Ms        2x2                     60  sym

Grand total is 21 elements using 362 bytes
```

2.1.5　符号运算机理和变量假设

1. 符号运算的工作机理

在 MATLAB 的默认安装下，MATLAB 的符号计算是由 MuPAD 引擎在其专有的内存工作空间中执行，而只是把计算结果送回到 MATLAB 的内存空间。

在 MATLAB 环境中，每当借助 sym 或 syms 指令定义一个带限定性假设的符号变量时，

就发生以下过程:

- 启动 MuPAD 引擎,并开启一个专供 MuPAD 使用的内存空间。
- 该定义变量被保存在 MATLAB 内存空间。(这是因为,在 MATLAB 空间中的变量仅仅是承接或转运计算结果的保留空间。)
- 对该定义变量的限定性假设(Assumption),则被保存在 MuPAD 的内存空间中,并对此后的 MuPAD 的工作方式进行约束。
  - 如果该定义变量不带任何约束,则 MuPAD 默认地在复数域里工作。
  - 如若该定义变量带 real 约束,则 MuPAD 只在实数域里工作。如此等等。

2. 对符号变量的限定性假设

在不对符号变量进行专门设置的情况下,MuPAD 符号计算总把变量默认为"复数变量"。在实际应用中,常常需要对符号变量进行各种假设(Assumptions for symbolic variables),如实数、正数等。对符号变量进行限定性假设借助 sym 或 syms 指令进行。具体格式如表 2.1-2所列。

表 2.1-2 sym 和 syms 指令可作的限定性假设

指令格式	含义	
x=sym('x')	定义"复数"符号标量 x	复数域是默认设置的数域
x=sym('x', 'real')	定义"实数"符号标量 x	
x=sym('x', 'positive')	定义"正数"符号标量 x	
syms x y z	定义"复数"符号标量 x, y, z	复数域是默认设置的数域
syms x y z real	定义"实数"符号标量 x, y, z	
syms x y z positive	定义"正数"符号标量 x, y, z	

说明

- 由以上格式可知,sym 或 syms 定义的变量只有三种,或复数,或实数,或正数。
- 对符号变量的限定可以借助 evalin 指令进行得更细腻(如偶数,区间等)。但所涉内容超出本书范围。

3. 清除变量和撤销假设

由于符号变量和其假设存放在不同的内存空间,因此删除符号变量和撤销关于变量的假设是两件需要分别处理的事。具体执行指令如下:

```
clear x                                    清除 MATLAB 内存中的 x 变量
syms x clear                               撤销 MuPAD 内存中对变量 x 的任何假设,而恢复为"复数"变量
evalin(symengine,'getprop(x)')             获取关于 x 的限定性假设
evalin(symengine,'anames(Properties)')     列出 MuPAD 内存中带限定假设的符号变量名
reset(symengine)                           重启 MuPAD 引擎,清空 MuPAD 内存中所有内容
```

说明

- clear x 指令仅仅删除 MATLAB 内存中的 x,并不改变 MuPAD 内存中可能已有"关于

x 的假设”。换句话说，如果以后运算中又重新出现 x 符号变量，那么“关于原有 x 的假设”仍将强制约束新出现的 x 变量。

- sym x clear 指令只是撤销 MuPAD 中关于变量 x 的假设，并没有删除 MATLAB 内存中的变量 x，也不改变 x 所保存的内容。

例【2.1-6】 syms 对变量所作限定性假设的影响。本例目的：符号变量的默认数域是复数域；限定性假设的设置方法及影响；如何观察当前对符号变量有何假设；撤销假设的方法。

(1) 在默认的复数域求根

```
syms x clear
f = x^3 + 4.75 * x + 2.5;
rf = solve(f,x)                              % 求出 f = 0 的全部根
rf =
                    - 1/2
1/4 - (79^(1/2) * i)/4
(79^(1/2) * i)/4 + 1/4

evalin(symengine,'getprop(x)')               % 获悉 MuPAD 内存中关于 x 的假设        <4>
ans =
C_
```

(2) 求实数根

```
syms x real                                  % 设 x 是实数
rfr = solve(f,x)                             % 求出 f = 0 的实根
rfr =
- 1/2
evalin(symengine,'getprop(x)')               % 获悉 MuPAD 内存中关于 x 的假设        <8>
ans =
R_
```

(3) 仅在 MATLAB 内存中删除 x，再解新方程 $x^2+x+5=0$

```
clear x                                      % 仅清除 MATLAB 内存的 x 变量
syms x
g = x^2 + x + 5;
rg = solve(g,x)                              % 此前 x 为实数的假设仍然其作用，所以导致无解
Warning: Explicit solution could not be found.
> In solve at 98
rg =
[ empty sym ]
```

(4) 须在 MuPAD 解除 x 为实数的假设再解

```
syms x clear                                 % 重新假设 x 为复数                      <13>
rg = solve(g,x)                              % 可得到一对共轭根
rg =
- (19^(1/2) * i)/2 - 1/2
  (19^(1/2) * i)/2 - 1/2
```

【2.1-7】 本例演示：符号变量的创建和清除；带限定假设的变量的创建和清除；MATLAB内存和MuPAD内存的显示；指令 evalin(symengine,'anames(Properties)')的功用。

(1) 双精度变量、一般符号变量、带限定符号变量的创建

```
clear all                    % 清空 MATLAB 内存空间
reset(symengine)             % 重启 MuPAD 引擎，清空 MuPAD 内存        <2>
Da = 1.2;Dw = 1/3;           % 定义双精度变量
syms sa sw sx sy sz          % 定义一般符号变量
syms A B positive            % 定义限定变量 A 和 B                      <5>
syms C real                  % 定义限定变量 C                           <6>
```

(2) MATLAB 内存中的变量及数据类型

```
whos                                                          %  <7>
  Name      Size            Bytes  Class     Attributes

  A         1x1                58  sym
  B         1x1                58  sym
  C         1x1                58  sym
  Da        1x1                 8  double
  Dw        1x1                 8  double
  sa        1x1                60  sym
  sw        1x1                60  sym
  sx        1x1                60  sym
  sy        1x1                60  sym
  sz        1x1                60  sym
```

(3) MATLAB 内存中的全部符号变量

```
syms                         % 显示当前 MATLAB 内存中的全部符号变量     <8>
    'A'    'B'    'C'    'sa'    'sw'    'sx'    'sy'    'sz'
```

(4) MuPAD 内存中所有带"限定性假设"的符号变量

```
evalin(symengine,'anames(Properties)')                        %  <9>
                             % 显示 MuPAD 内存中各限定分属的变量名
ans =
{A, B, C}
```

(5) 清除"变量"操作的影响

```
clear A         % 在 MATLAB 内存中清除"带限定的符号变量 A"
syms            % 显示当前 MATLAB 内存中的全部符号变量，可发现 A 已消失    <11>

    'B'    'C'    'ans'    'sa'    'sw'    'sx'    'sy'    'sz'

evalin(symengine,'anames(Properties)')                           <12>
                % 显示 MuPAD 内存中各限定分属的变量名，发现 A 的限定假设并没有清除
ans =
{A, B, C}
```

(6) 清除"限定假设"操作的影响

```
syms B clear    % 清除关于 B 的"限定假设"                                <13>
```

```
syms                  % 显示当前 MATLAB 内存中的全部符号变量，可见 B 的存在未受影响

    'B'    'C'    'ans'    'sa'    'sw'    'sx'    'sy'    'sz'

evalin(symengine,'anames(Properties)')                                  %      <15>
                      % 显示当前 MuPAD 内存中各限定分属的变量名，关于 B 的限定假设已消失
ans =
{A, C}
```

2.1.6 符号帮助体系

现在，MATLAB 关于符号指令的帮助系统，已与 2008 年前有很大的不同。这是因为从那时起 MATLAB 不再把 Maple 作为符号计算引擎的缘故。

现在的符号运作帮助体系包含如表 2.1－4 所示的"三个层次"的求助方式：

表 2.1－4　符号运作的帮助体系

指　　令	含义及功用	参考节次及例题
help SymName	在此 SymName 代表待求助的符号计算指令名（如 laplace 等）	本节；例 2.1－8
doc mfunlist	在 MATLAB 的帮助浏览器中，展现"所有可调用的特殊函数及其调用格式"。	本节；例 2.1－8
doc(symengine)	引出 MuPAD 的帮助浏览器。在此环境中，可搜索 MuPAD 的任何函数、指令及其调用格式。	本节；例 2.1－8

(1) "直接调用符号计算指令"的求助

● 可求助的符号指令范围

不依赖 mfun，evalin，feval 等指令调用，而可在 MATLAB 环境中直接运作的符号计算指令，如 sin，inv，eig，solve，dsolve，laplace 等，都可采用这种求助方式。

欲获得此类全部指令的列表，可按如下操作步骤进行：在 MATLAB 的帮助浏览器的 Contents 页，展开 Symbolic Math Toolbox 节点，再点击其下的 Function Reference 节点即可。

● 求助方式

■ 像 MATLAB 普通指令那样，在 MATLAB 指令窗中借助 help 指令获得帮助。如在指令窗中，运行以下指令，可获得关于 lapalce 指令的使用说明：

```
help laplace
```

■ 像 MATLAB 普通指令那样，可利用 MATLAB 帮助浏览器获得相应的帮助。

(2) 借助 mfun 调用的"特殊函数指令"的求助

● 可求助的符号指令范围

凡依赖 mfun 指令调用才能被执行的“特殊函数指令”。

● 求助方式

■ 在 MATLAB 指令窗中运行 doc mfunlist，或利用 MATLAB 帮助浏览器搜索栏对 mfunlist 搜索，都可得到关于特殊函数指令调用格式的较完整帮助信息。假如想更深入地了解特殊函数的使用要领，就必须进入 MuPAD 帮助浏览器（见第三种求助方式相关叙述）。

■ 在 MATLAB 指令窗中直接运行 help mfunlist，只能得到关于函数名、输入量等的简单列表。而关于调用格式的信息很少。

(3) 借助 evalin 或 feval 调用的“MuPAD 指令”的求助。

● 可求助的符号指令范围

所有 MuPAD 提供的计算函数、指令。注意：在 MATLAB 环境中，服务于符号计算的有些指令（如 sym 等）并不属于 MuPAD，而仅存在于 MATLAB 环境中。因此，这类指令不能在 MuPAD 中获得帮助。

● 求助步骤

■ 首先，在指令窗中，运行 doc(symengine)，引出 MuPAD 专用帮助浏览器。

■ 在 MuPAD 帮助浏览器中，搜索所需求助的函数或指令名。

例 **【2.1-8】** 关于 laplace，erfc，Ei，rec 四个指令的求助过程。本例目的：通过具体指令表述求助的步骤。

(1) 检查指令是否“可直接调用的符号计算指令”

● 搜索法（此法适于指令名已知情况）

■ 方法一：在 MATLAB 帮助浏览器左上方的搜索栏中输入待查询函数名 laplace，就可见到如图 2.1-1 所示的界面。

■ 方法二：在 MATLAB 指令窗中，直接运行 doc laplace 或 help laplace，也能得到很好的帮助信息。

● 函数表法（此法适于通览）

在 MATLAB 帮助浏览器左侧 Contents 页，点击＜Symbolic Math Toolbox/Functions＞节点。

(2) 检查指令是否“借助 mfun 调用的符号计算指令”

● 搜索法（适于指令名已知情况）

■ 方法一：在 MATLAB 帮助浏览器左上方的搜索栏中输入待查询函数名 erfc(或 Ei)，就可见到如图 2.1-2 所示的界面。（注意：不要误把数值计算指令 erfc 当成符号计算指令，因为这两种指令都存在。）

● 函数表法（适于查阅和通览）

■ 方法一：在 MATLAB 帮助浏览器左侧 Contents 页，点击＜Symbolic Math Toolbox/Functions/Special Functions＞节点，就能引出如图 2.1-2 所示的全部特殊函数列表。这已经是 MATLAB 帮助浏览器中提供的全部信息。假如读者仍感觉简单，则要借助 MuPAD 帮助浏览器才能实现。

■ 方法二：在 MATLAB 帮助浏览器的搜索栏中输入 mfunlist 关键词，也能引出如图 2.1-2 所示的全部特殊函数列表。

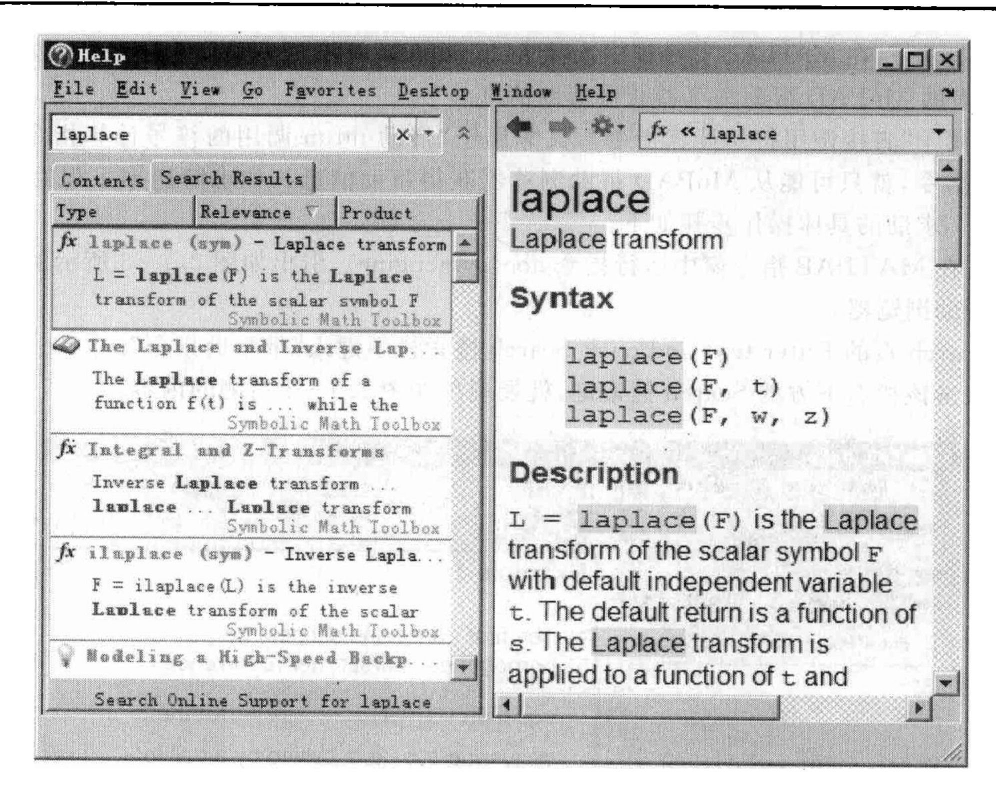


图 2.1－1　“可直接调用的符号计算指令”的帮助信息搜索

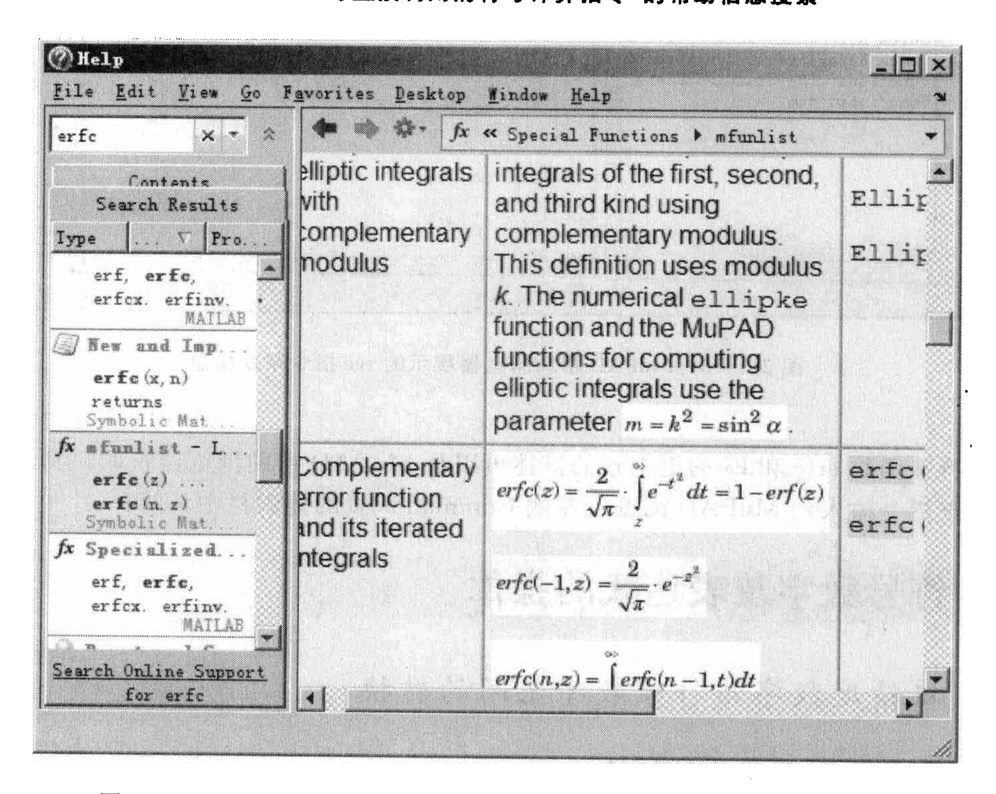


图 2.1－2　MATLAB 帮助浏览器展示全部特殊函数符号计算指令及用法

■ 方法三：在 MATLAB 指令窗中，运行 help mfunlist，只能得到最简单的特殊函数名列表。

(3) 借助 MuPAD 帮助浏览器获得帮助信息

既非属于"直接调用符号计算指令"，又非属于"借助 mfun 调用的符号计算指令"的那种符号计算指令，就只可能从 MuPAD 帮助浏览器获得帮助信息。如求解递推方程的 rec 就是这种指令。求助的具体操作步骤如下：

- 先在 MATLAB 指令窗中运行指令 doc(symengine)，引出如图 2.1-3 所示的 MuPAD 帮助浏览器。
- Search 页的 Enter text and click Search 搜索栏中键入"带帮助指令"rec。
- 点击该栏右下方的 Search 搜索键，就展现出如图 2.1-3 右侧的内容。

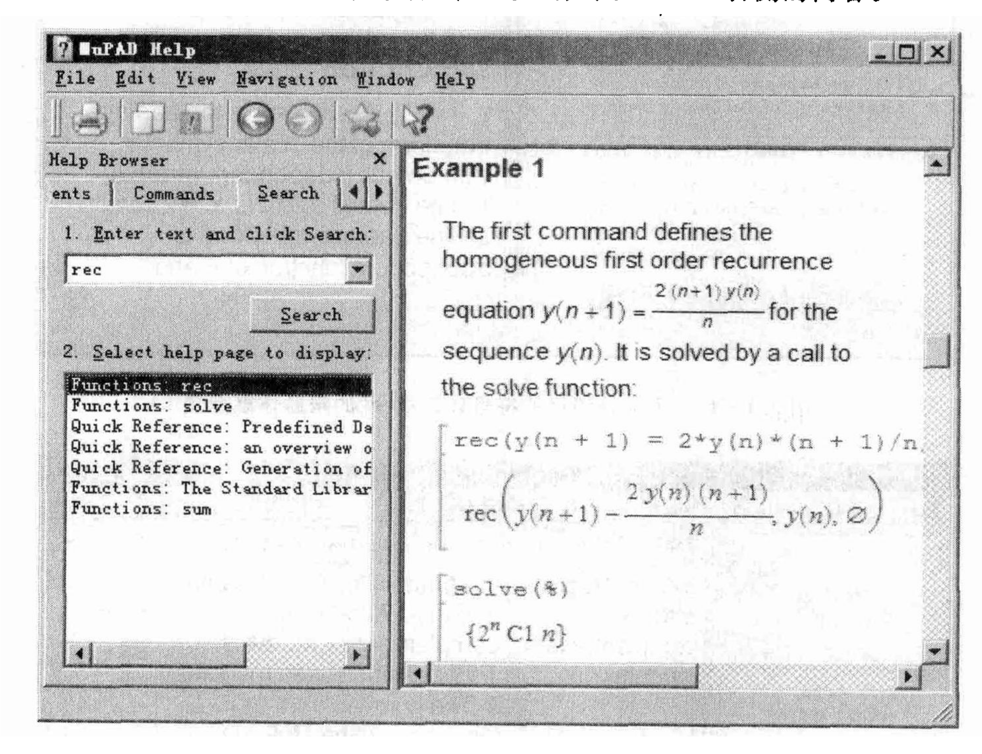


图 2.1-3 MuPAD 帮助浏览器展示的 rec 指令帮助信息

说明

关于特殊函数 erfc 和 Ei 的更细致的描述可以从 MuPAD 帮助浏览器查取。具体执行时，只要把待查指令输入到 MuPAD 浏览器左侧 Commands 页的搜索栏中，即可。

2.2 符号数字及表达式的操作

2.2.1 双精度数字与符号数字之间的转换

1. 双精度数字向符号数字的转换

假如需要借助 sym 函数把双精度类数字转换为符号数字时，那么有以下 5 种形式可

供选择：

sym(Num, ' r ')	双精度数字 Num 的"有理分数"表达的符号数字
sym(Num)	sym(Num, ' r ')的简略形式
sym(Num, ' f ')	双精度数字 Num 的"两个二进制数"近似表达符号数字
sym(Num, ' e ')	数值类数字 Num 的带估计误差的"广义有理表达"符号数字
sym(Num, ' d ')	数值类数字 Num 的"十进制浮点"近似表达符号数字

说明

- 以上 5 种调用格式中，第一个输入量 Num 是"双精度数字"。
- sym(Num, ' r ') 的输出是有理分数表达的符号数字。即通过 p/q , p * 2^q , n^(p/q)形式表示的符号数(在此，n, p, q 都是整数)。
- sym(Num, ' f') 给出形如$\pm[2^e+N\times2^{(-52+e)}]$表示的符号数。其中 e 为整数，N 为非负整数。
- sym(Num, 'e') 给出两项：一项为广义有理表达；另一项为 Num 理论有理表达与 Num 实际机器浮点表达之间的估计误差。
- sym(Num, 'd') 给出"十进制浮点"近似表示的有效数字位数，受 digits 指令控制。在默认情况下，即在 digits(32)控制下，sym(Num, ' d ') 给出 Num 的 32 位"十进制浮点"近似表达。
- 顺便提醒读者注意：以上 5 条指令的第一输入量 Num 的形式与第 5.1.1 节 sym('Num') 指令输入'Num' 的差别。
  - 'Num' 是字符串数字。当 'Num' 用作符号计算函数的输入量时，它体现数字的理论真值。
  - Num 是字面表示数字。在 MATLAB 环境中，实际上是以双精度(16 位相对精度)的近似方式保存的。换句话说，当 Num 用作符号计算函数的输入量时，它体现的是字面理论数字的双精度近似。
- 事实上，在符号运算中，"双精度数字"都会自动地按 sym(Num, ' r ')格式转换为符号数字。

2. 符号数字向双精度数字转换

MATLAB 的数值计算和可视化指令(如泛函指令 quad，绘图指令 plot)不接受符号数字，而只接受数值类数字。在这种情况下，就必须采用以下指令进行数据类型转换。

double(Num sym)	把符号数字 Num_sym 转换为双精度数字

说明

- 符号数字有时与双精度数字在外观上相似。此时可以借助 class 辨别数据类型。
- 值得提醒注意：指令格式 double('Num')将把字符串数字 ' Num ' 转换为字符的 ASCII 码值数组。

2.2.2　符号数字的任意精度表达形式

数值计算与符号计算间的最重要区别在于：数值计算一定存在截断误差，且在计算过程中不断传播，而产生累积误差；符号计算，其运算过程是在完全准确情况下进行的，不产生累积误

差。符号计算的这种准确性是以降低计算速度和增加内存需求为代价换来的。为了兼顾计算精度和速度，MATLAB还针对符号计算提供一种“变精度”算法。具体指令如下。

digits	显示当前环境下符号数字“十进制浮点”表示的有效数字位数
digits(n)	设定符号数字“十进制浮点”表示的有效数字位数
xs=vpa(x)	据表达式 x 得到 digits 指定精度下的符号数字 xs
xs=vpa(x,n)	据表达式 x 得到 n 位有效数字的符号数字 xs

说明

- 变精度函数 vpa(x)的运算精度受它之前运行的 digits(n)控制。MATLAB 对 digits 指令的默认精度设置是 32 位。
- vpa(x,n)只在运行的当时起作用。
- x 可以是符号对象，也可以是数值对象，但指令运作后所得结果 xs 一定是符号数字。

例【2.2-1】 本例演示：digits 的用法及影响；vpa 指定符号数字有效位的影响和含义；符号计算引擎的重置指令 reset(symengine)。

(1) 重置符号计算引擎，产生准确符号数字

```
reset(symengine)                        % 重新启动符号计算引擎                <1>

sa = sym('1/3 + sqrt(2)')               % 定义准确符号数
sa =
2^(1/2) + 1/3
```

(2) 采用默认设置“变精度算法”的计算结果，及有效数位的含义

```
digits                                  % 观察当前“变精度算法”的有效数位。
                                                                              <3>

Digits = 32


format long
a = 1/3 + sqrt(2)                       % 定义双精度数
sa_Plus_a = vpa(sa + a,20)              % 给出 20 位有效数字结果              <6>
sa_Minus_a = vpa(sa - a,20)             % 指定 20 位有效数字，但显示出 36 位数字，  <7>
                                        % 因为小数点后的 16 个 0 不是有效位。
a =
   1.747546895706428
sa_Plus_a =
3.4950937914128567869
sa_Minus_a =
 - 0.000000000000000022658064826339973669
```

(3) digits 设置“数位”和 vpa 中指定“数位”的不同影响

```
sa32 = vpa(sa)                          %                                     <8>
digits(48)                              % 把其后的“变精度运算”设置为 48 位有效数字  <9>
sa5 = vpa(sa,5)                         % 该设定仅影响 sa5 数位，对其后无影响。   <10>
```

```
sa48 = vpa(sa)                          % 仍产生由48位有效数字的十进制符号数    <11>
sa32 =
1.7475468957064283821350220557543
sa5 =
1.7475
sa48 =
1.74754689570642838213502205754303141190300520871
```

说明

- 第⟨1⟩条指令的作用是重置符号计算引擎，可彻底消除所有关于符号变量已作假设的影响。
- 第⟨6⟩⟨7⟩指令的运行结果有利于理解 digits 和 vpa 所指定的有效数字位数的含义。
- 本例中指令⟨3⟩和指令⟨6⟩ ～ ⟨11⟩对 digits 和 vpa 进行了交叉设置，目的是表现它们各自对有效数位的不同影响。

2.2.3 符号表达式的基本操作

符号运算中有许多操作指令，如 collect(合并同类项)、expand(对指定项展开)、factor(进行因式或因子分解)、horner(转换成嵌套形式)、numden(提取公因式)、simplify(恒等式简化)、pretty(习惯方式显示)、coeffs(获取符号多项式系数)等，其中最常用的是：

simple(EXPR)　　运用包括 simplify 在内的各种指令把 EXPR 转换成最简短形式

说明

EXPR 可以是符号表达式或矩阵。在这种情况下，这些指令将对该矩阵的元素逐个进行操作。

例【2.2-2】 简化 $f=\sqrt[3]{\frac{1}{x^3}+\frac{6}{x^2}+\frac{12}{x}+8}$。重点演示：simple 的用法。

```
syms x
f = (1/x^3 + 6/x^2 + 12/x + 8)^(1/3)
g1 = simple(f)
f =
(12/x + 6/x^2 + 1/x^3 + 8)^(1/3)
g1 =
((2 * x + 1)^3/x^3)^(1/3)
```

2.2.4 表达式中的置换操作

1. 公因子法简化表达

符号计算结果显得烦冗的一个重要原因是：有些子表达式会多次出现在不同地方。为了使表达式简洁易读，MATLAB 提供了如下指令。

RS=subexpr(S)	从 S 中自动提取公因子 sigma，并把采用 sigma 重写的 S 赋给 RS
RS=subexpr(S,'w')	从 S 中自动提取公因子，记它为 w，并把采用 w 重写的 S 赋给 RS
[RS,w]=subexpr(S,'w')	该调用格式的效果与 RS=subexpr(S,'w')相同。

说明

S可以使符号表达式,符号表达式矩阵。当S是符号表达式矩阵时,subexpr所提取的公因子是对整个符号矩阵而言的,而不是对矩阵中一个个元素表达式给出的。

例【2.2-3】 对符号矩阵 $A=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ 进行特征向量分解。本例演示:复杂符号矩阵的公因子法简化表达;指令subexpr的正确使用格式。

(1) 符号矩阵的特征值和特征向量分解

```
clear                          % 清空所有内存变量
A = sym('[a b;c d]')           % 经字符串直接定义符号矩阵
[V,D] = eig(A)                 % 符号矩阵的特征值、特征向量分解。
A =
[ a, b]
[ c, d]
V =
[(a/2 + d/2 - a^2 - 2 * a * d + d^2 + 4 * b * c)^(1/2)/2)/c - d/c, (a/2 + d/2 + (a^2 - 2 * a * d + d^2 + 4 * b * c)^(1/2)/2)/c - d/c]
[                                          1,                    1]
D =
[a/2 + d/2 - (a^2 - 2 * a * d + d^2 + 4 * b * c)^(1/2)/2,                    0]
[                          0, a/2 + d/2 + (a^2 - 2 * a * d + d^2 + 4 * b * c)^(1/2)/2]
```

(2) 表达式中公因式的自动识别

```
subexpr([V;D])       % 由subexpr的最简单调用格式,可得到公因式            <4>
who                  % 由此指令结果可知:subexpr([V;D])运行后产生sigma和ans
sigma =
(a^2 - 2 * a * d + d^2 + 4 * b * c)^(1/2)
ans =
[ (a/2 + d/2 - sigma/2)/c - d/c, (a/2 + d/2 + sigma/2)/c - d/c]
[                    1,                    1]
[        a/2 + d/2 - sigma/2,                    0]
[                    0,        a/2 + d/2 + sigma/2]

Your variables are:

A      D      V      ans     sigma
```

(3) 对D进行简化,而"自动提取的公因式"的名称是指定的

```
Dw = subexpr(D,'w')      % 把自动提取的公因式记为w,而Dw是用w重记D后的表达
w =
(a^2 - 2 * a * d + d^2 + 4 * b * c)^(1/2)
Dw =
[ a/2 + d/2 - w/2,             0]
[            0, a/2 + d/2 + w/2]
```

(4) 对V,D同时简化,而"自动提取的公因式"的名称是指定的

```
[RVD,w] = subexpr([V;D],'w')        % 给出合成矩阵[V;D]的公因式表达方式        <7>
RVD =
[ (a/2 + d/2 - w/2)/c - d/c, (a/2 + d/2 + w/2)/c - d/c]
[                         1,                         1]
[           a/2 + d/2 - w/2,                         0]
[                         0,           a/2 + d/2 + w/2]
w =
(a^2 - 2*a*d + d^2 + 4*b*c)^(1/2)
```

说明

- 在 subexpr 指令的所有调用格式中，公因式是机器自动寻找的，而不能指定的。
- 由于指令〈4〉和指令〈7〉中被重写的是 [V;D]，因此所得的重写结果 ans 及 RVD 需要正确阅读：重写结果 ans 或 RVD 的上 2 行子阵是特征向量阵；下 2 行是特征值阵。

2. 通用置换指令

RES=subs(ES,old,new)　　用 new 置换 ES 中的 old 后产生 RES

RES=subs(ES,new)　　用 new 置换 ES 中的自由变量后产生 RES

说明

- ES 可以是符号表达式、符号数组，old 可以是表达式、变量、或变量胞元数组，new 可以是变量、数字、或胞元数组等。
- subs 指令的输出结果的属性取决于 new 的属性。只要 ES 中有符号对象没被置换，所得结果就保持"符号类型"不变。

例【2.2-4】 本例目的：用简单算例演示 subs 的各种置换方式；演示符号计算与数值计算之间的一种转换途径。

(1) 产生符号函数

```
clear
syms a b x;
f = a*sin(x) + b
f =
b + a*sin(x)
```

(2) 符号表达式置换，得到新的符号表达式

```
f1 = subs(f,sin(x),'log(y)')        % subs 的第三输入量可以使"字符串"        <4>
class(f1)                           % 为观察结果的类型而设
f1 =
b + a*log(y)
ans =
sym
```

(3) 单个变量被双精度数字置换，结果依然是符号对象

```
f2 = subs(f,a,3.11)                 % 双精度数字 3.11 先被 sym(3.11)处理后再代入
class(f2)                           % 为观察结果的类型
f2 =
```

```
b + (311 * sin(x))/100
ans =
sym
```

(4) 所有变量被数字(至少含一个符号数字)置换,结果依然是符号对象

```
f3 = subs(f,{a,b,x},{2,5,sym('pi/3')})          % 注意胞元数组的用法          <8>
class(f3)
f3 =
3^(1/2) + 5
ans =
sym
```

(5) 所有变量被双精度数值置换,结果是双精度数字

```
format                                          % 为恢复对双精度数字显示的默认设置
format compact                                  % 为使在 Notebook 中显示紧凑
f4 = subs(f,{a,b,x},{2,5,pi/3})                 % 注意胞元数组的用法          <12>
class(f4)
f4 =
    6.7321
ans =
double
```

(6) 一个变量被双精度数组置换,得到符号数组

```
f5 = subs(f,x,0:pi/2:pi)
class(f5)
f5 =
[ b, a + b, b]
ans =
sym
```

(7) 所有变量被双精度数(其中一个是数组)置换,得到双精度数组

```
t = 0:pi/10:2 * pi;
f6 = subs(f,{a,b,x},{2,3,t})                    % 注意胞元数组的用法          <17>
plot(t,f6)                                      % 绘图 2.2 - 1
f6 =
  Columns 1 through 8
    3.0000    3.6180    4.1756    4.6180    4.9021    5.0000    4.9021    4.6180
  Columns 9 through 16
    4.1756    3.6180    3.0000    2.3820    1.8244    1.3820    1.0979    1.0000
  Columns 17 through 21
    1.0979    1.3820    1.8244    2.3820    3.0000
```

(8) 通过两次置换获得双精度数组

```
k = (0.5:0.1:1)';
f6 = subs(subs(f,{a,b},{k,2}),x,t);             % 注意两次置换                <20>
size(f6)                                        % 观察 f6 数组大小
plot(t,f6)                                      % 绘图 2.2 - 2
```

```
ans =
     6    21
```

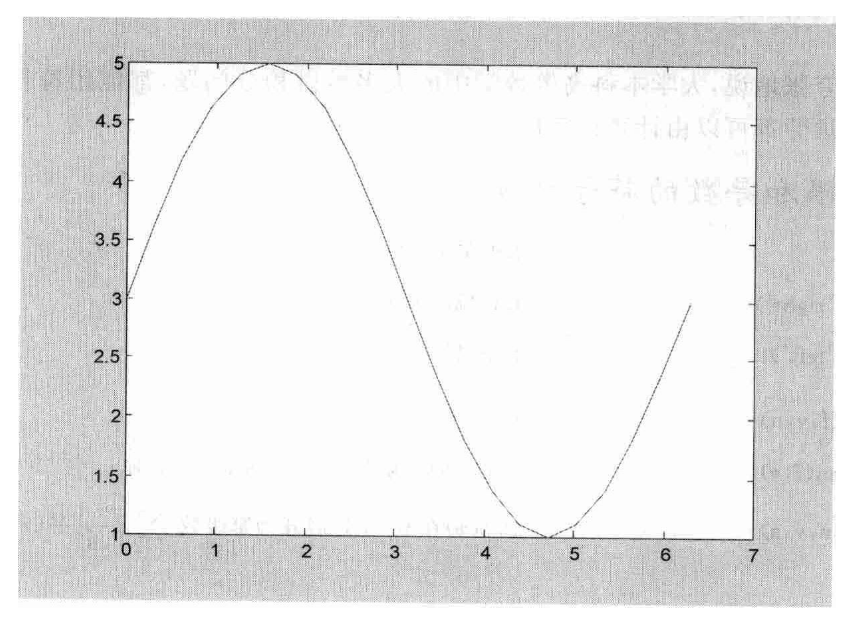


图 2.2-1　利用符号表达式变量置换产生的单根曲线

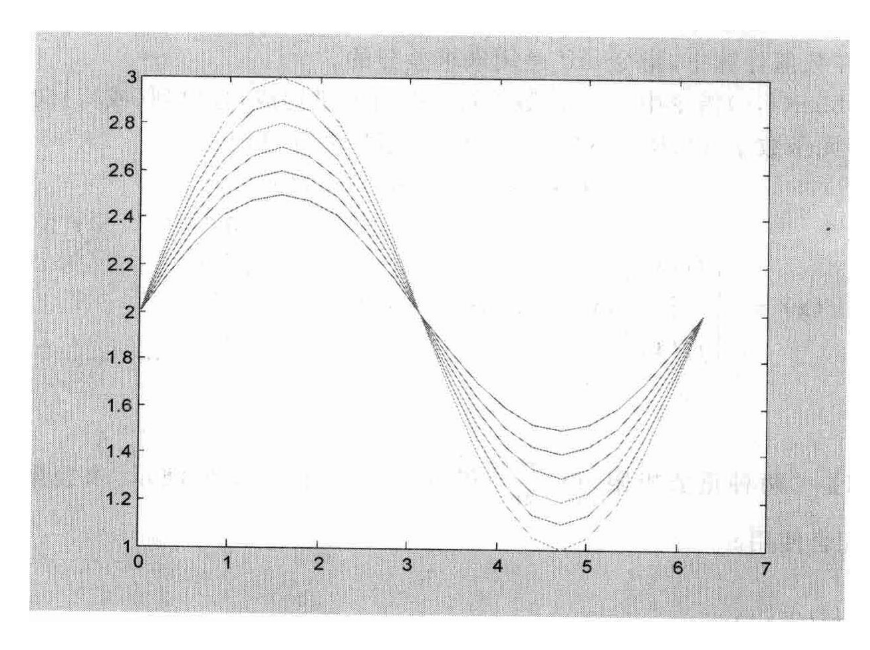


图 2.2-2　利用两次 subs 置换产生的多根曲线

说明

- 指令 subs(ES,old,new)中的 old 可以取：串表达式（见指令〈4〉）；符号变量（见指令〈8〉）；胞元数组（见指令〈8〉〈12〉〈17〉〈20〉中花括号的使用）。
- 值得指出：指令〈17〉〈20〉 演示了 subs 指令在符号表达式向双精度数值计算转换中的功用。

2.3 符号微积分

可以毫不夸张地说，大学本科高等数学中的大多数微积分问题，都能用符号计算解决，手工笔算演绎的烦劳都可以由计算机完成。

2.3.1 极限和导数的符号计算

limit(f,v,a)	求极限 $\lim\limits_{v \to a} f(v)$
limit(f,v,a,'right')	求右极限 $\lim\limits_{v \to a^+} f(v)$
limit(f,v,a,'left')	求左极限 $\lim\limits_{v \to a^-} f(v)$
dfdvn=diff(f,v,n)	求 $\dfrac{d^n f(v)}{dv^n}$
fjac=jacobian(f,v)	求多元向量函数 $\boldsymbol{f}(v)$ 的 jacobian 矩阵
r=taylor(f,n,v,a)	把 $f(v)$ 在 $v=a$ 处展开为幂级数 $\sum\limits_{k=0}^{n-1} \dfrac{f^{(k)}(a)}{k!}(x-a)^k$

说明

- f 是矩阵时，求极限和求导操作对元素逐个进行，但自变量定义在整个矩阵上。
- x 省缺时，自变量会自动由 symvar 确认；n 省缺时，默认 n=1。
- 注意：在数值计算中，指令 diff 是用来求差分的。
- 在 jacobian(f,v) 指令中，v 由函数 f 的所有自变量构成，且以列（或行）向量形式出现。
- 向量多元函数 $f(\boldsymbol{x}): R^m \to R^n$ 在 x_0 的线性近似展开可记为

$$f(\boldsymbol{x}) \approx f(x_0) + J(x_0)(x - x_0)$$

式中，$f(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) \\ \vdots \\ f_n(\boldsymbol{x}) \end{bmatrix}$，$\boldsymbol{x} = [x_1, \cdots, x_m]$；$J(\boldsymbol{x}) = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_m} \\ \vdots & & \vdots \\ \dfrac{\partial f_n}{\partial x_1} & \cdots & \dfrac{\partial f_n}{\partial x_m} \end{bmatrix}$ 称为 Jacobian 矩阵。

例【2.3-1】 两种重要极限 $\lim\limits_{t \to 0} \dfrac{\sin kt}{kt}$ 和 $\lim\limits_{x \to \infty} \left(1 - \dfrac{1}{x}\right)^{kx}$。本例演示：求极限指令的使用；subs，vpa 的配合使用。

```
syms t x k
s = sin(k * t)/(k * t);
f = (1 - 1/x)^(k * x);
Lsk = limit(s,0)                          % 机器确定自由变量 t 趋于 0 时函数 s 的极限
Ls1 = subs(Lsk,k,1)                       % 给出 Sa(t)|t=0 的值
Lf = limit(f,x,inf)                       % 指定自由变量 x 趋于"正无穷"时 f 函数的极限
Lf1 = vpa(subs(Lf,k,sym('-1')),48)        % 给出 48 位精度的自然数
Lsk =
1
```

```
Ls1 =
1
Lf =
1/exp(k)
Lf1 =
2.7182818284590452353602874713526624977572470937
```

说明

- 在数学上，$\mathrm{sinc}_\pi(x)=\frac{\sin\pi x}{\pi x}$是著名的归一化 Sinc 函数。$\mathrm{sinc}_\pi(t-k)$可在 $L^2(R)$函数空间中构成带限函数的正交基，因此在数字信号处理和通信理论中常见此函数。
- 在数学上，$\mathrm{sinc}(x)=\frac{\sin x}{x}$称为"非归一化 Sinc 函数"。而它在"信号与系统"等教科书中，又常记为 $Sa(t)\doteq\frac{\sin t}{t}$，称为抽样函数。

例【2.3-2】　$f=\begin{bmatrix} a & t^3 \\ t\cos x & \ln x \end{bmatrix}$，求$\frac{\mathrm{d}f}{\mathrm{d}x},\frac{\mathrm{d}^2 f}{\mathrm{d}t^2},\frac{\mathrm{d}^2 f}{\mathrm{d}t\mathrm{d}x}$。本例演示：求导运算是对矩阵元素逐个进行的；求一阶导数、高阶导数、混合导数。

```
syms a t x;
f = [a,t^3;t * cos(x), log(x)];
df = diff(f)                          % 求矩阵 f 对 x 的导数
dfdt2 = diff(f,t,2)                   % 求矩阵 f 对 t 的二阶导数
dfdxdt = diff(diff(f,x),t)            % 求二阶混合导数
df =
[           0,    0]
[ - t * sin(x), 1/x]
dfdt2 =
[ 0, 6 * t]
[ 0,     0]
dfdxdt =
[          0, 0]
[ - sin(x), 0]
```

例【2.3-3】　求 $f(x_1,x_2)=\begin{bmatrix} x_1 e^{x_2} \\ x_2 \\ \cos(x_1)\sin(x_2) \end{bmatrix}$的 Jacobian 矩阵$\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} \end{bmatrix}$。本例演示：jacobian 指令的用法。

```
syms x1 x2;
f = [x1 * exp(x2);x2;cos(x1) * sin(x2)];     % (3 * 1)向量函数
v = [x1;x2];                  % (2 * 1)自变量。即便写 v = [x1,x2]，结果一样。    <3>
Jf = jacobian(f,v)            % (3 * 2)的 Jacobian 矩阵
```

```
Jf =
[          exp(x2),        x1 * exp(x2)]
[                0,                   1]
[ - sin(x1) * sin(x2), cos(x1) * cos(x2)]
```

说明

指令<3>把自变量 x1 和 x2 写成列向量 v。但若写成 v=[x1,x2],所得 Jacobian 矩阵完全一样。

例【2.3-4】 $f(x)=\sin|x|$,求 $f'_x(0)$,$f'_x(x)$。本例演示:(A) 在理论层面上,对问题本身的分析,导数的极限定义,区间端点处的导数;(B) 在符号计算层面上,subs 的变量置换用法,limit 的左极限、右极限用法,diff, legend, char 的用法;(C) 数值绘图指令如何用符号计算结果绘制曲线(见图 2.3-1)。

观察 $f(x)$ 可知:除 $x=0$ 处存疑外,$f(x)$ 是处处光滑可导的。

(1) 对于 $x\geqslant 0$,据导数定义求右导数

```
clear
syms x
syms d positive
f_p = sin(x);                                          % 据 x≥0 时,由 sin|x|改写而成
df_p = limit((subs(f_p,x,x+d) - f_p)/d,d,0)            % 求 x>0 区间的导数        <5>
df_p0 = limit((subs(f_p,x,d) - subs(f_p,x,0))/d,d,0)   % x = 0+ 的右导数          <6>
df_p =
cos(x)
df_p0 =
1
```

(2) 对于 $x\leqslant 0$,据导数定义求左导数

```
f_n = sin( - x);                                         % 据 x≤0 时,由 sin|x|改写而成
df_n = limit((f_n - subs(f_n,x,x - d))/d,d,0)            % 求 x<0 区间的导数       <8>
df_n0 = limit((subs(f_n,x,0) - subs(f_n,x, - d))/d,d,0)  % x = 0- 的左导数         <9>
df_n =
 - cos(x)
df_n0 =
 - 1
```

(3) 直接利用 diff 求导数

```
f = sin(abs(x));
dfdx = diff(f,x)                                   %  <11>
dfdx0 = subs(dfdx,x,0)         % x = 0 处给出错误的导数值    <12>
dfdx =
cos(abs(x)) * sign(x)
dfdx0 =
     0
```

(4) 图形观察(图 2.3-1)

```
clf
```

```
xn = - 3/2 * pi:pi/50:0;xp = 0:pi/50:3/2 * pi;xnp = [xn,xp(2:end)];
hold on
plot(xnp,subs(f,x,xnp),'k','LineWidth',3)                          % <14>
plot(xn,subs(df_n,x,xn),'--r','LineWidth',3)
plot(xp,subs(df_p,x,xp),':r','LineWidth',3)
legend(char(f),char(df_n),char(df_p),'Location','NorthEast')       % <17>
grid on
xlabel('x')
hold off
```

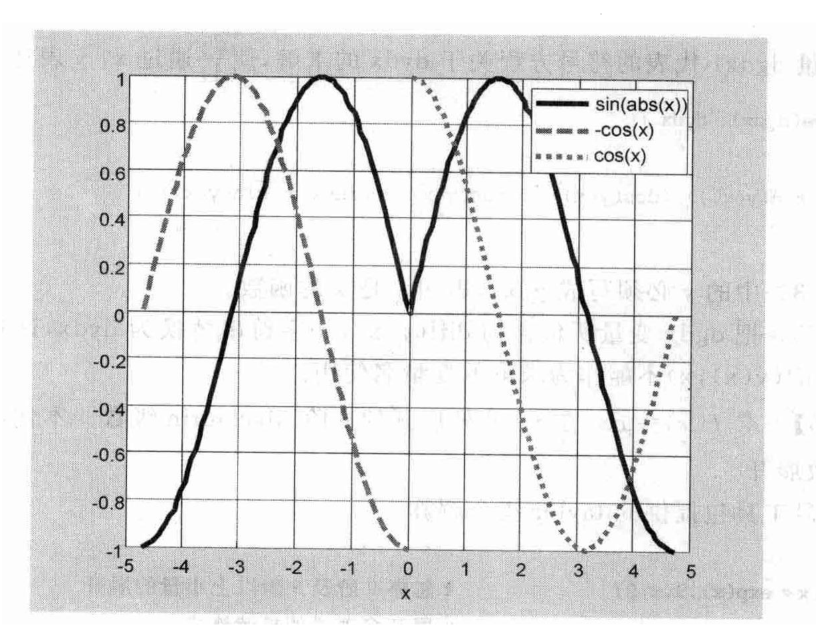


图 2.3-1　函数及其导函数

说明

- 通过分区间求左右导数，可知导函数是处处光滑可导的。具体的导函数如下

$$f'_x = \begin{cases} \cos x & x > 0 \\ -\cos x & x < 0 \end{cases} \quad \text{或写为 } f'_x = \text{sign}(x) \cdot \cos x \qquad x \neq 0$$

- 注意：指令〈11〉直接求导，给出了 $f'_x = \text{sign}(x) \cdot \cos x$ 结果，而没有指明该式的成立条件，这是软件设计缺陷。
- 指令〈5〉〈6〉〈8〉〈9〉〈14〉〈15〉〈16〉演示了 subs 的三种典型用法：表达式置换；数值标量置换；数值数组置换。
- 指令〈17〉中，使用了 char 指令把符号对象转变字符串，这是 legend 指令所要求的。

例【2.3-5】 设 $\cos(x + \sin y) = \sin y$，求 $\frac{dy}{dx}$。本例演示：如何实现隐函数求导。

(1) 对方程（即隐函数）求导

```
clear
syms x
g = sym('cos(x + sin(y(x))) = sin(y(x))')                          % <3>
```

```
dgdx = diff(g,x)                              % 对方程求导                  <4>
g =
cos(x + sin(y(x))) = sin(y(x))
dgdx =
- sin(x + sin(y(x))) * (cos(y(x)) * diff(y(x), x) + 1) = cos(y(x)) * diff(y(x), x)
```

(2) 用符合规则的新变量名 dydx 替代 dgdx 中的 diff(y(x),x)

```
dgdx1 = subs(dgdx,'diff(y(x),x)','dydx')      % 必须采取的步骤              <5>
dgdx1 =
- sin(x + sin(y(x))) * (dydx * cos(y(x)) + 1) = dydx * cos(y(x))
```

(3) 对变量 dgdx1 代表的符号方程关于 dydx 的求解,使$\frac{dy}{dx}$通过 x, y 表达出来

```
dydx = solve(dgdx1,'dydx')
dydx =
- sin(x + sin(y(x)))/(cos(y(x)) + cos(y(x)) * sin(x + sin(y(x))))
```

说明

- 指令<3>中的 y 必须写成 y(x),表明 y 是 x 的函数。
- 指令<5>把 dgdx 变量所包含的 diff(y(x),x)字符串替换为 dydx,这是必须的步骤,因为 diff(y(x),x)不能作为求解的变量名使用。

例【2.3-6】 求 $f(x)=xe^x$ 在 $x=0$ 处展开的 8 阶 Maclaurin 级数。本例演示:一元函数的 Taylor 级数展开。

(1) 用符号工具包提供的 taylor 指令解算

```
syms x
r = taylor(x * exp(x),9,x,0)        % 忽略 9 阶及 9 阶以上小量的展开          <2>
pretty(r)                           % 展开多项式的易读格式
r =
x^8/5040 + x^7/720 + x^6/120 + x^5/24 + x^4/6 + x^3/2 + x^2 + x


    8      7      6     5     4     3
   x      x      x     x     x     x      2
  ---- + --- + --- + -- + -- + -- + x  + x
  5040   720    120    24    6     2
```

(2) 直接调用 MuPAD 引擎解算

```
R = evalin(symengine,'series(x * exp(x),x = 0,8)')                      %     <4>
pretty(R)                                % 展开多项式的易读格式
R =
x + x^2 + x^3/2 + x^4/6 + x^5/24 + x^6/120 + x^7/720 + x^8/5040 + O(x^9)


              3     4     5     6      7       8
      2      x     x     x     x      x       x          9
 x + x  + -- + -- + -- + --- + --- + ---- + O(x  )
              2     6    24    120    720    5040
```

说明

- 本例列出 Taylor 级数展开的两个不同指令。指令 evalin 下例还将用到。
- 多变量函数的 Taylor 展开,请见例 2.3-7 和例 2.8-4。
- pretty 指令把“通常在一个物理行里显示的表达式”扩展成“用多个物理行显示的表达式”。为的是便于阅读。

例【2.3-7】 求 $\sin(x^2+y)$在 $x=0,y=0$ 处的截断 8 阶小量的 Taylor 展开近似。本例演示:二元函数的 Taylor 展开。

```
TL1 = evalin(symengine,'mtaylor(sin(x^2 + y),[x,y],8)')
TL1 =
(x^6 * y^2)/12 - x^6/6 + (x^4 * y^3)/12 - (x^4 * y)/2 - (x^2 * y^6)/720 + (x^2 * y^4)/24 - (x^2 * y
^2)/2 + x^2 - y^7/5040 + y^5/120 - y^3/6 + y

class(TL1)
ans =
sym
```

说明

- taylor 指令可比较方便地用于一元函数展开,或二元函数关于其一个变量的展开。本例调用库函数 mtaylor 对二元函数进行 Taylor 级数展开。

- evalin 指令进入 MuPAD 空间,调用 MuPAD 所具有的函数指令 mtaylor 进行符号计算。关于 evalin 的其他信息,请看第 2.1.5 节和第 2.8.3 节。

2.3.2 序列/级数的符号求和

对于数学上用通式表达的级数(Series)求和的问题,即 $\sum_{v=a}^{b} f(v)$,可用 MATLAB 的求和指令解决。具体如下:

s=symsum(f,v,a,b) 求通式 f 在指定变量 v 取遍$[a,b]$中所有整数时的和。

说明

- f 是矩阵时,求和对元素逐个进行,但自变量定义在整个矩阵上。
- v 省缺时,f 中的自变量由 findsym 自动辨认;b 可以取有限整数,也可以取无穷大。
- a, b 可同时省缺,此时默认求和的自变量区间为 [0,v-1]。

例【2.3-8】 求 $\sum_{k=1}^{n}\frac{1}{k(k+1)}$, $\sum_{k=1}^{\infty}\frac{x^{2k-1}}{2k-1}$, $\sum_{k=1}^{\infty}\left[\frac{1}{(2k-1)^2},\frac{(-1)^k}{k}\right]$。本例演示:symsum 的用法。

(1) 有限项级数之和

```
syms n k
f1 = 1/(k * (k + 1));
s1 = symsum(f1,k,1,n)
s1 =
```

```
1 - 1/(n + 1)
```

(2) 无限项级数之和

```
f2 = x^(2 * k - 1)/(2 * k - 1);
s2 = symsum(f2,k,1,inf)
s2 =
piecewise([abs(x) < 1, atanh(x)])
```

(3) 符号通式数组求级数和

```
f3 = [1/(2 * k - 1)^2,( - 1)^k/k];
s3 = symsum(f3,k,1,inf)
s3 =
[ pi^2/8, - log(2)]
```

说明

- 通式中的自变量只取整数值。
- 求和指令中的 f 可以是符号矩阵，此时求和操作将对矩阵中的"元素通式"逐个进行，但通式矩阵的自变量及其取值区间对各"元素通式"是相同的。
- 由于 symvar 对 f 符号矩阵的自变量进行确认时，不是对该矩阵元素分别进行的，而是把矩阵 f 作为一个统一的对象处理的。所以当 f 矩阵中含有一个以上非常数符号变量时要特别注意，求和对哪个符号变量进行。

2.3.3 符号积分

积分 $F(x)=\int f(x)\mathrm{d}x$，就是要找 $f(x)$ 的原函数(Antiderivative) $F(x)$，使 $\frac{\mathrm{d}F(x)}{\mathrm{d}x}=f(x)$。一般说来，无论哪种积分(不定积分、定积分等)都比微分更难求取。

与数值积分相比，符号积分的优点是：指令简单，适应性强。缺点是：计算用时可能很长，给出的符号积分结果可能是冗长而生疏的"闭"符号表达式，甚或完全不能给出"封闭"解。

在相当多的情况下，当"参数积分限"用具体数值替代时，符号积分将能给出具有"任意精度"的定积分值。

求积分指令的具体使用格式如下：

intf=int(f,v)	给出 f 对指定变量 v 的(不带积分常数的)不定积分
intf=int(f,v,a,b)	给出 f 对指定变量 v 的定积分。

说明

- 与 symsum ，diff 指令一样，当 f 是矩阵时，积分将对元素逐个进行。
- v 省缺时，积分对 symvar 确认的变量进行。
- a，b 分别是积分的下、上限，允许它们取任何值或符号表达式。

例【2.3-9】 不定积分 $\int(x\ln x)\mathrm{d}x$ 。本例演示：积分指令的使用；计算结果的简化。

```
clear
syms a b x
f1 = x * log(x)
s1 = int(f1,x)
```

```
s1 = simple(s1)                        % 对积分结果简化                    <5>
f1 =
x * log(x)
s1 =
(x^2 * (log(x) - 1/2))/2
s1 =
x^2 * (log(x)/2 - 1/4)
```

例【2.3-10】 求 $\int\begin{bmatrix} ax & bx^2 \\ \frac{1}{x} & \sin x \end{bmatrix}\cdot \mathrm{d}x$ 。本例演示:积分指令对符号函数矩阵的作用。

```
f2 = [a * x,b * x^2;1/x,sin(x)]
disp(' ')
disp('The integral of f is')
pretty(int(f2))
f2 =
[ a * x,  b * x^2]
[ 1/x, sin(x)]

The integral of f is

  + -                       - +
  |      2          3         |
  |   a x        b x          |
  |   ----,      ----         |
  |    2          3           |
  |                           |
  |  log(x), - cos(x)         |
  + -                       - +
```

说明

pretty 指令把“通常在一个物理行里显示的表达式”扩展成“用两个物理行显示的表达式”。目的是读起来更方便。

例【2.3-11】 求三重积分 $\int_1^2\int_{\sqrt{x}}^{x^2}\int_{\sqrt{xy}}^{x^2y}(x^2+y^2+z^2)\mathrm{d}z\mathrm{d}y\mathrm{d}x$ 。本例演示:内积分上下限都是函数的情况。

```
syms x y z
F2 = int(int(int(x^2 + y^2 + z^2,z,sqrt(x * y),x^2 * y),y,sqrt(x),x^2),x,1,2)
VF2 = vpa(F2)                        % 积分结果用 32 位数字表示
Warning: Explicit integral could not be found.
F2 =
(14912 * 2^(1/4))/4641 - (6072064 * 2^(1/2))/348075 + (64 * 2^(3/4))/225 + 1610027357/6563700
VF2 =
224.92153573331143159790710032804675767707137601 2
```

说明

对于内积分上下限为函数的多重积分，如若采用数值方法求取，那数值计算的编程将很不轻松。

例【2.3-12】 求阿基米德(Archimedes)螺线 $r=a\cdot\theta,(a>0)$在 $\theta=0$ 到 φ 间的曲线长度函数，并求出 $a=1,\varphi=2\pi$ 时的曲线长度。本例演示：如何进行曲线积分；符号变量的属性限定；subs, diff, int, vpa 的综合使用；绘图指令 ezplot 的参变量绘图法，及改变曲线色彩的方法。

据数学分析知，$x=r\cos\theta, y=r\sin\theta, \mathrm{d}l=\sqrt{(x'_\theta)^2+(y'_\theta)^2}\mathrm{d}\theta$ 为弧长元素，而曲线长度 $L(\varphi)=\int_0^\varphi\sqrt{(x'_\theta)^2+(y'_\theta)^2}\mathrm{d}\theta$。

(1) 求曲线长度函数

```
syms a r theta phi
r = a * theta;                                  % 螺线定义
x = r * cos(theta);                             % x 的极坐标
y = r * sin(theta);                             % y 的极坐标
dLdth = sqrt(diff(x,theta)^2 + diff(y,theta)^2);
warning off                                     % 为抑制提示性警告而设              <6>
L = simple(int(dLdth,theta,0,phi))                                             % <7>
L =
(phi * (a^2 * phi^2 + a^2)^(1/2) + log(phi + (phi^2 + 1)^(1/2)) * (a^2)^(1/2))/2
```

(2) $a=1,\varphi=2\pi$ 时的曲线长度

```
L_2pi = subs(L,[a,phi],sym('[1,2 * pi]'))        % 获得完全准确值
L_2pi_vpa = vpa(L_2pi)                           % 计算 32 精度近似值
L_2pi =
log(2 * pi + (4 * pi^2 + 1)^(1/2))/2 + pi * (4 * pi^2 + 1)^(1/2)
L_2pi_vpa =
21.256294148209098800702512272566108823470931047 6
```

(3) 螺线 $r=a\cdot\theta$ 和螺线长度函数 $L=L(\varphi)$ 的绘制(图 2.3-2)

```
L1 = subs(L,a,sym('1'));                         % 使螺线长度函数中，参数 a 数字化
ezplot(L1 * cos(phi),L1 * sin(phi),[0,2 * pi])   % 画螺线长度函数曲线
grid on                                          % 给坐标纸打方格线
hold on                                          % 在上述图形窗内，可继续画图。
x1 = subs(x,a,sym('1'));                         % 使螺线 x 坐标参数 a 数字化
y1 = subs(y,a,sym('1'));                         % 使螺线 y 坐标参数 a 数字化
h1 = ezplot(x1,y1,[0,2 * pi]);                   % 画螺线，并产生图柄              <16>
set(h1,'Color','r','LineWidth',5)                % 通过图柄，改变曲线图形对象属性
title(' ')                                       % 消除自动写出的图名              <18>
legend('螺线长度-幅角曲线','阿基米德螺线')
hold off                                         % 在上述图形窗中，不再允许画任何图形。
```

说明

- 与使用 Maple 引擎相比，MuPAD 引擎执行指令〈7〉需要的时间可能较长。读者实践时，要有耐心。

- 假如没有指令〈6〉,运行过程中会给出警告信息。
- 在 $L=L(\varphi)$ 函数中,L 是角度为 φ 时螺线的长度,在极坐标上用"幅值"表示。
- 第〈18〉条指令用来消除 ezplot 自动写出的图名。这样做的原因是:在本例中,由第〈16〉条指令写出的图名,不能准确表达该图上的两条曲线。

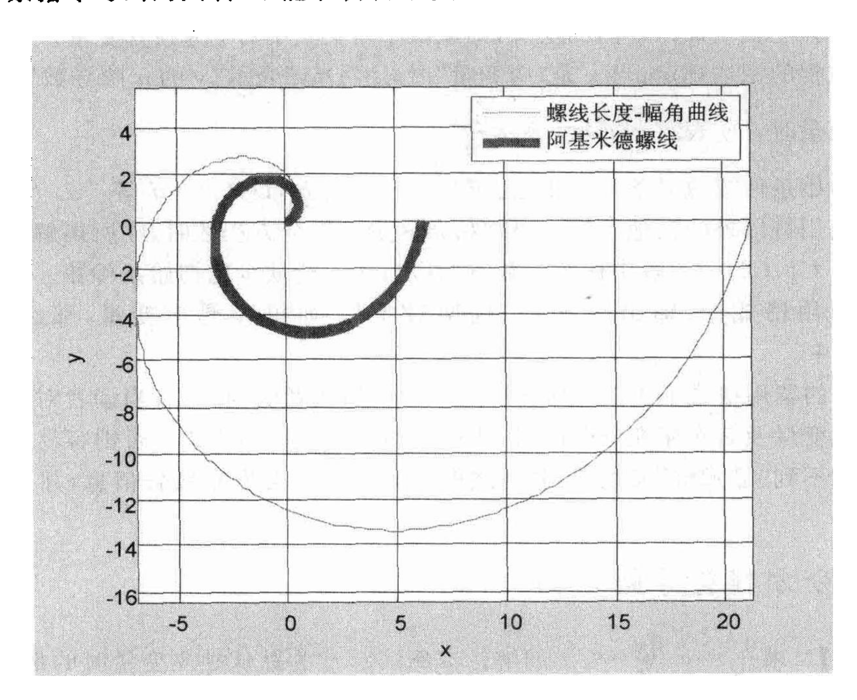


图 2.3-2 阿基米德螺线(粗红线)和螺线长度函数(细蓝线)

2.4 微分方程的符号解法

2.4.1 符号解法和数值解法的互补作用

从数值计算角度看,与初值问题求解相比,微分方程边值问题的求解显得复杂和困难。对于应用数学工具去解决实际问题的科研人员来说,此时,不妨通过符号计算指令进行求解尝试。因为,对于符号计算来说,不论是初值问题,还是边值问题,其求解微分方程的指令形式相同,且相当简单。

当然,符号计算可能花费较多的机时,可能得不到简单的解析解,可能得不到封闭形式的解,甚至也可能无法求解。

不管怎样,既然没有万能的微分方程一般解法,那么请记住:求解微分方程的符号法和数值法有很好的互补作用。

2.4.2 求微分方程符号解的一般指令

求解符号微分方程最常用的指令格式为如下两种:

```
S=dsolve('eq1, eq2, …, eqn', 'cond1, cond2, …, condn', 'v')
```

S=dsolve('eq1', 'eq2', …, 'eqn', 'cond1', 'cond2', …, 'condn', 'v')

说明

- 输入量包括三部分:微分方程、初始条件、指定独立变量。其中微分方程是必不可少的输入内容。其余视需要而定,可有可无。输入量必须以字符串形式编写。
- 若不对独立变量加以专门的定义,则默认小写英文字母 t 为独立变量。
- 微分方程的记述规定:当 y 是“应变量”时,用“Dny”表示“y 的 n 阶导数”。在 t 为默认独立变量时,Dy 表示$\frac{dy}{dt}$;Dny 表示$\frac{d^n y}{dt^n}$。
- 关于初始条件或边界条件的规定:应写成 $y(a)=b$,$Dy(c)=d$ 等。a, b, c, d 可以是变量使用符以外的其他字符。当初始条件少于微分方程数时,在所得解中将出现任意常数符 C1,C2,…。解中任意常数符的数目等于所缺少的初始条件数。
- 在本调用格式中,输出量 S 是“构架对象”。如果 y 是应变量,那么关于它的解在S.y 中。
- dsolve 的调用格式不止以上所介绍的一种。但编者认为,只有当读者对微分方程组默认独立变量及其次序充分理解时,才适于使用其他调用格式。否则容易引起混乱。
- 在既找不到“显式解”又找不到“隐式解”的情况下,会发布警告信息,并且 S 为空符号对象。

2.4.3 微分方程符号解示例

例【2.4-1】 求$\frac{dx}{dt}=y,\frac{dy}{dt}=-x$ 的解。本例演示:t 为默认独立变量时的最简单调用;输出量的格式;clear all 对微分方程解中任意常数序号的影响。

```
clear all                        % 为读者运行该例所得结果中任意常数序号与本书一致         <1>
S = dsolve('Dx = y,Dy = - x')    % 从它的输出,只能看到 2 个“域”S.x 和 S.y
disp(' ')
disp(['微分方程的解',blanks(2),'x',blanks(22),'y'])
disp([S.x,S.y])
S =
    x: [1x1 sym]
    y: [1x1 sym]

微分方程的解  x                      y
[ C2 * cos(t) + C1 * sin(t), C1 * cos(t) - C2 * sin(t)]
```

说明

- 这是 dsolve 一种较简略的调用格式。由 symvar 自动确定输出“应变量”为 x 和 y,默认独立变量是 t。C1,C2 是任意常数。
- 微分方程解中任意常数 C 的序号,并不总是以 1, 2, 3 等依次出现的。它的编号与此前 MATLAB 解算其它微分方程的运行历程有关。
- 指令〈1〉的存在与否,对微分方程解的正确性没有任何影响。但是,在任何 MATLAB 运行历程后,读者若希望解中任意常数序号与本书一致,那末指令〈1〉是有用的。

顺便指出：clear all 的使用，会降低计算速度，假如此前解算过同类问题的话。

● 在写微分方程时，最好遵循“导数在前函数在后，导数阶数降阶”的次序。否则有可能运行出错。

【例2.4-2】 图示微分方程 $y=xy'-(y')^2$ 的通解和奇解的关系。本例演示：微分方程的书写规则；独立变量的特别指定，存在多解时的输出量形式；绘图指令 ezplot，get，set，title 的配套使用及字体控制。

（1）解微分方程

```
clear all                              % 为方程解中任意常数与本书一致而设          <1>
y = dsolve('(Dy)^2 - x * Dy + y = 0','x')  % 注意书写规则                          <2>
y =
        x^2/4
 C3 * x - C3^2
```

（2）画“解”曲线

```
clf,hold on                            % 允许在同一图轴上画多条曲线
hy1 = ezplot(y(1),[-6,6,-4,8],1);      % 画奇解 y(1) = x^2/4,并记录图柄 hy1       <4>
set(hy1,'Color','r','LineWidth',5)     % 把奇解画成粗红线
for k = -2:0.5:2                       % 画通解 y(2) = C3 * x - C3^2              <6>
    y2 = subs(y(2),'C3',k);            % 为画一组线,对不定常数 C5 赋不同值 k       <7>
    ezplot(y2,[-6,6,-4,8],1)
end                                                                              % <9>
hold off                               % 不再在此图轴上绘画
box on                                 % 产生封闭型图轴框
legend('奇解','通解','Location','Best')
ylabel('y')
title(['\fontsize{14}微分方程',' (y '')^2    xy '' + y = 0 ','的解'])
```

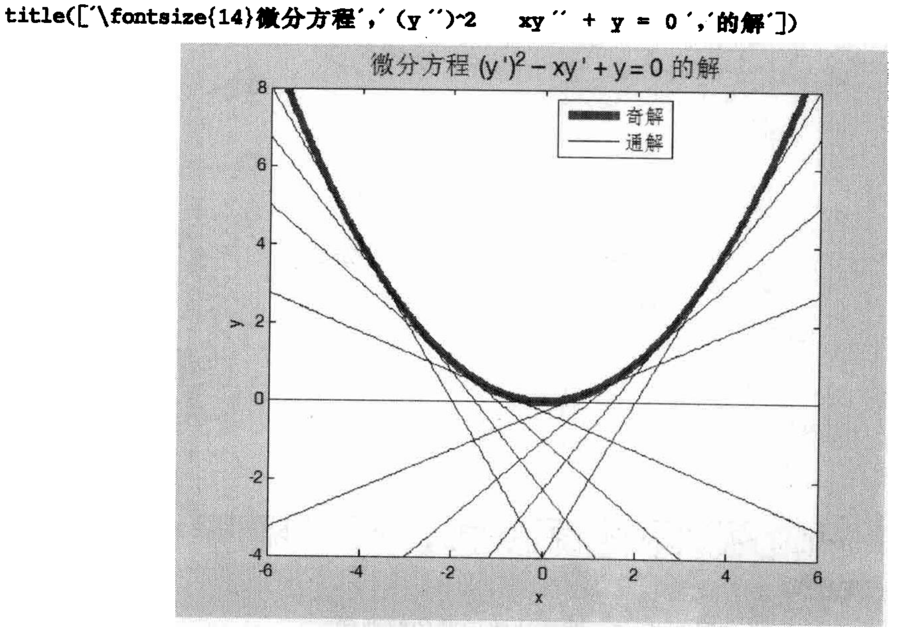


图 2.4-1 通解和奇解曲线

说明

- dsolve 有求通解和奇解的能力。本例奇解是通解直线族的包络抛物线。
- ezplot 对线没有色彩、线型控制力，但总可通过 get，set 对图柄的操作实现。如本例指令〈4〉产生图柄 hy1，然后再由 set 指令对线色、线粗加以修饰。
- ezplot 不能同时画多条曲线，而必须用循环解决。如指令〈6〉到〈9〉就为多条线。
- 关于本例指令〈7〉的两点特别说明：
  - 指令〈7〉中被替换常数之所以写 'C3'，是通过观察指令〈2〉计算结果而知的。
  - 指令〈7〉的 subs 指令中，被替换的变量之所以采用“字符串”形式 'C3' 定义，是因为 y 是一个“元符号表达式”。也就是说，y 表达式中的所有变量名（包括 C3 在内）并不真实地存在于在内存中。
  - 指令〈7〉可用以下两条指令取代

```
syms C3;y2 = subs(y(2),C3,k);
```

例【2.4-3】 求解两点边值问题：$xy''-3y'=x^2$，$y(1)=0$，$y(5)=0$。本例演示：dsolve 解边值问题；可视化微分方程的解；ezplot 和 plot 的混合使用。

（1）求解边值问题

```
y = dsolve('x * D2y - 3 * Dy = x^2','y(1) = 0,y(5) = 0','x')
y =
(31 * x^4)/468 - x^3/3 + 125/468
```

（2）观察“解”的图形

```
xn = -1:6;
yn = subs(y,'x',xn)          % 自变量 x 用数组替代(这组数字供不同算例比较用)
ezplot(y,[-1,6])
hold on
```

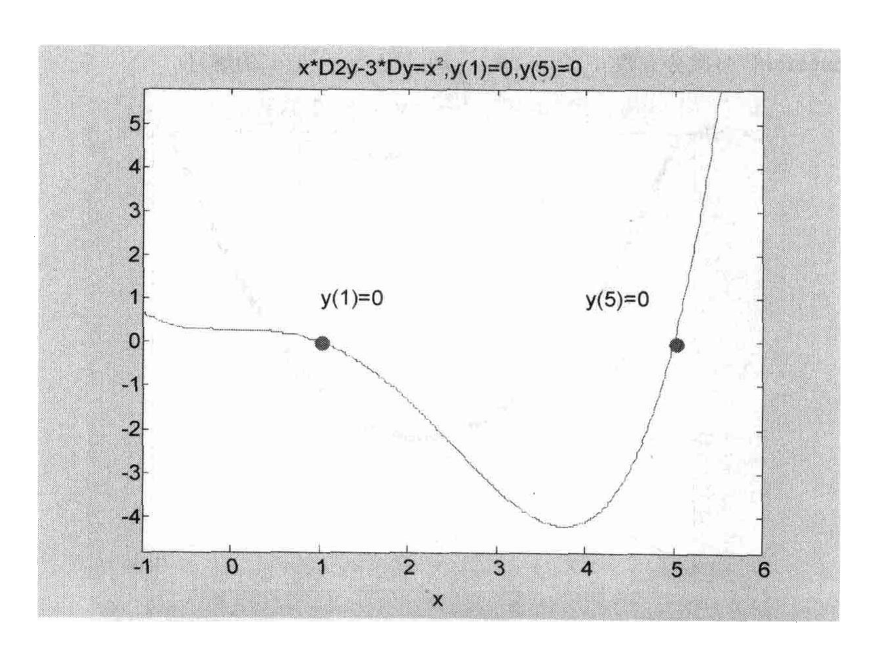


图 2.4-2 两点边值问题的解曲线

```
plot([1,5],[0,0],'.r','MarkerSize',20)
text(1,1,'y(1) = 0')
text(4,1,'y(5) = 0')
title(['x * D2y  -  3 * Dy  =  x^2',',    y(1) = 0,y(5) = 0'])
hold off
yn =
    0.6667     0.2671          0    -1.3397    -3.3675    -4.1090     0.0000    14.1132
```

说明

- 与数值解法相比，符号法求解微分方程的优点是程序编写比较简单；缺点是对微分方程的适应性较差。换句话说，有许多微分方程，也许是符号法无法解决的或求解时间过长。
- 对于本科教材上的微分方程，符号法也许是比较适当的解法。

2.5 符号变换和符号卷积

Fourier 变换、Laplace 变换、Z 变换和卷积在信号处理和系统动态特性研究中起着重要作用。本节将讨论这些变换和卷积符号算法实现。

2.5.1 Fourier 变换及其反变换

时域中的 $f(t)$ 与它在频域中的 Fourier 变换 $F(\omega)$ 之间存在如下关系：

$$F(\omega) = \int_{-\infty}^{\infty} f(t)\mathrm{e}^{-\mathrm{j}\omega t}\mathrm{d}t \tag{2.5-1}$$

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)\mathrm{e}^{\mathrm{j}\omega t}\mathrm{d}\omega \tag{2.5-2}$$

由计算机完成这种变换的途径有两条：一是，直接调用指令 fourier 和 ifourier 进行；二是，根据式(2.5－1)和式(2.5－2)定义，利用积分指令 int 实现。下面只介绍指令 fourier 和 ifourier 的使用及注意事项。至于据定义求变换，请读者自己尝试，想必不会有什么困难。

Fw=fourier(ft,t,w)　　求"时域"函数 ft 的 Fourier 变换 Fw

ft=ifourier(Fw,w,t)　　求"频域"函数 Fw 的 Fourier 反变换 ft

说明

- ft 是以 t 为自变量的"时域"函数；Fw 是以圆频率 w 为自变量的"频域"函数。
- 在此给出的是 fourier，ifourier 指令的完整调用格式。虽然它们都有默认调用格式，但本书建议：在版本特性不清楚的情况下，用户应慎用默认调用格式，以免出错。
- Fourier 变换(Fourier Transformation)以法国数学、物理学家 Jean Baptiste Joseph Fourier(1768 — 1830)命名。

例【2.5－1】 求单位阶跃函数的 Fourier 变换。本例演示：fourier 指令能正确执行"广义 Fourier"变换；fourier，ifourier 指令的正确使用；MATLAB 中单位阶跃函数 heaviside(t)的具体定义；阶跃函数曲线中间断点的绘制。

(1) 求 Fourier 变换

```
syms t w
```

```
ut = heaviside(t);
UT = fourier(ut)
UT =
pi * dirac( - w) - i/w
```

(2) 求 Fourier 反变换进行验算

```
Ut = ifourier(UT,w,t)                    % 反变换验算
SUt = simple(Ut)                         % 进一步简化                          <5>
Ut =
(pi + pi * (2 * heaviside(t) - 1))/(2 * pi)
SUt =
heaviside(t)
```

(3) 单位阶跃函数曲线

```
t = - 2:0.01:2;
ut = heaviside(t);
kk = find(t = = 0);                                  % t = 0 的元素下标            <8>
plot(t(kk),ut(kk),'.r','MarkerSize',30)              % t = 0 处画点
hold on
ut(kk) = NaN;                                        % 为使 t = 0 处画线断开        <10>
plot(t,ut,' - r','LineWidth',3)
plot([t(kk),t(kk)],[ut(kk - 1),ut(kk + 1)],'or','MarkerSize',10)
                                                     % 画下连线右端和上连线左端
hold off
grid on
axis([ - 2,2, - 0.2,1.2])
xlabel('\fontsize{14}t'),ylabel('\fontsize{14}ut')
title('\fontsize{14}Heaviside(t)')
```

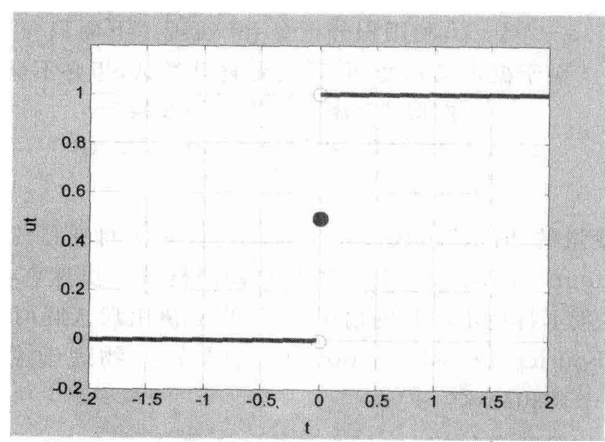


图 2.5 - 1 Heaviside(t)定义的单位阶跃函数

说明

- 指令〈5〉的使用得当。
- 本例曲线间断点绘制中，指令〈8〉〈10〉是关键。

● 在 MATLAB 中,单位阶跃函数 heavisideh(x)的数学定义是

$$H(x)=\begin{cases}0 & x<0\\ 0.5 & x=0\\ 1 & x>0\end{cases}$$

● 单位阶跃函数是由英国电气工程师、数学物理学家 Oliver Heaviside (1850 − 1925) 首先提出并应用。

例【2.5-2】 利用 Heaviside 函数构成矩形脉冲 $y=\begin{cases}A & |t|<\tau/2\\ 0 & |t|>\tau/2\end{cases}$ 的 Fourier 变换。本例演示:heaviside 的调用;变换、反变换;simplify 和 simple 指令的适当运用;绘图指令的配用。

(1)求 Fourier 变换

```
syms A t w tao
yt = A * (heaviside(t + tao/2) - heaviside(t - tao/2));          % 定义矩形脉冲
Yw = fourier(yt,t,w)                                              % Fourier 变换
Yw_fy = simplify(Yw)                                              % 恒等式法简化
Yw_fy_e = simple(Yw_fy)                                           % 最短式简化指令恰当
Yw =
A * ((1/exp((tao * w * i)/2)) * ( - pi * dirac( - w) + i/w) - exp((tao * w * i)/2) * ( - pi * dirac
( - w) + i/w))
Yw_fy =
(2 * A * sin((tao * w)/2))/w
Yw_fy_e =
(2 * A * sin((tao * w)/2))/w
```

(2) 用反变换验算

```
Yt = ifourier(Yw_fy_e,w,t)                     % 反变换
Yt_e = simple(Yt)                              % 指令运用不当,没改变"隐式"            <7>
Yt =
(A * transform::fourier(sin((tao * w)/2)/w, w, t))/pi
Yt_e =
- A * (heaviside(t - tao/2) - heaviside(t + 1/2 * tao))
```

(3) 时域曲线(见图 2.5-2)绘制(设 A=1,tao=3)

```
t3 = 3;
tn = - 3:0.1:3;
yt13 = subs(yt,{A,tao},{1,t3})
yt13n = subs(yt13,'t',tn);
kk = find(tn == - t3/2|tn == t3/2);              % 找到 t 等于"正负 tao/2"的元素下标     <13>
plot(tn(kk),yt13n(kk),'.r','MarkerSize',30)      % 画间断点
yt13n(kk) = NaN;                                 % 目的是使连线在"正负 tao/2"处断开     <15>
hold on
plot(tn,yt13n,'-r','LineWidth',3)
hold off
grid on
axis([-3,3,-0.5,1.5])
```

```
yt13 =
heaviside(t + 3/2) - heaviside(t - 3/2)
```

（4）频域曲线（见图 2.5－3）绘制

```
Yw13 = subs(Yw_fy_e,{A,tao},{1,t3});
subplot(2,1,1),ezplot(Yw13),grid on
subplot(2,1,2),ezplot(abs(Yw13)),grid on
```

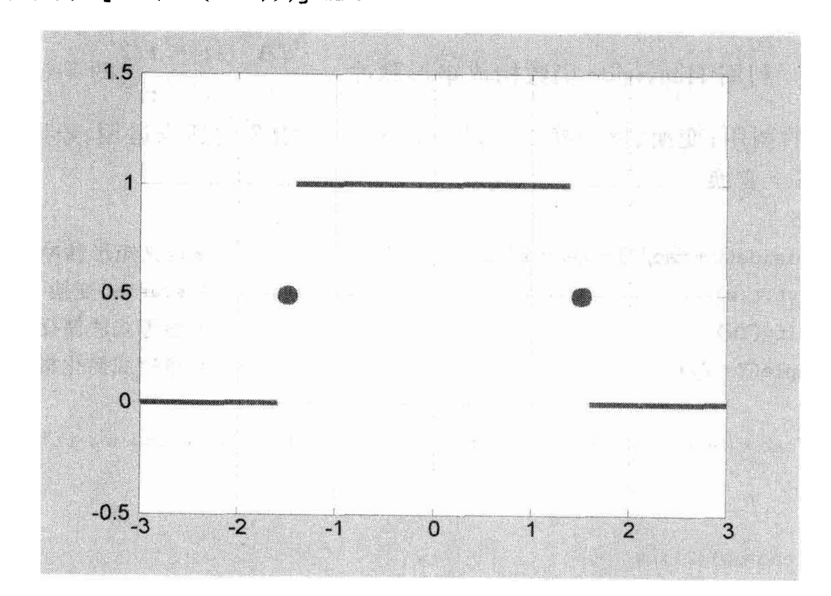


图 2.5－2　由 Heaviside(t)构造的矩形波

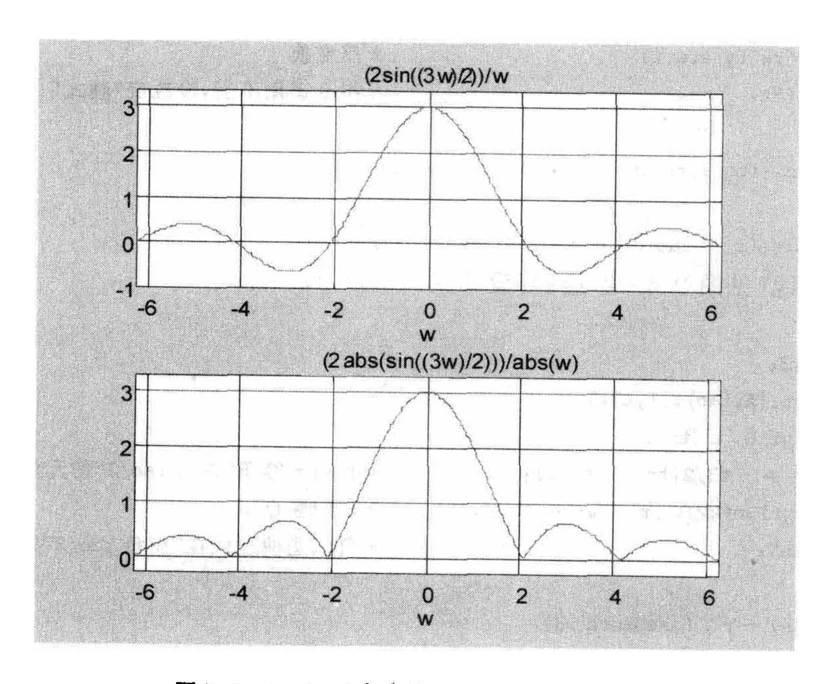


图 2.5－3　矩形脉冲的频率曲线和幅度频谱

说明

- simplify 和 simple 指令运用是否恰当，无一般规律可循，只能通过尝试解决。
- 指令〈12〉find 指令的用法具有典范性。
- 本例和前一例都展示了绘制间断曲线应采取的措施。这些处理方法具有典范性。
- 若根据定义，直接通过积分求本例矩形脉冲的 Fourier 变换，那将是十分费时的。在使用 MuPAD 引擎时，感觉尤其明显。

例【2.5-3】 求 $f(t)=\begin{cases} e^{-(t-x)} & t \geqslant x \\ 0 & t < x \end{cases}$ 的 Fourier 变换，在此 x 是参数，t 是时间变量。本例演示：借助 Heaviside 函数正确定义因果函数；十分谨慎使用 fourier 的不完整调用格式；在被变换函数中包含多个符号变量的情况下，对被变换的自变量给予指明，可保证计算结果的正确。

(1) 正确定义被变换的函数

```
clear
syms t x w
ft = exp( - (t - x)) * heaviside(t - x);     % 构造因果函数 f(t)                <3>
gt = exp( - (t - x));                        % 是非因果函数                     <4>
```

(2) fourier 指令完整调用格式

```
F1 = simple(fourier(ft,t,w))                 % 对因果函数变换，给出题目所需结果
G1 = simple(fourier(gt,t,w))                 % 产生不同结果
F1 =
(1/exp(w * x * i))/(1 + w * i)
G1 =
transform::fourier(exp(x - t), t, - w)
```

(2) fourier 指令不完整的调用格式产生错误

```
F2 = simple(fourier(ft,t))                   % 误把 x 当作时间变量，又误把 t 当作频率变量
F3 = simple(fourier(ft))                     % 误把 x 当作时间变量
F2 =
- exp(t^2 * i)/( - 1 + t * i)
F3 =
- (1/exp(t * w * i))/( - 1 + w * i)
```

说明

- 在此，因果函数是指：在某时刻前，时间函数值一直为零的函数。
- 本例指令〈3〉构造的函数符合题意。而指令〈4〉产生的函数是非因果的。

2.5.2 Laplace 变换及其反变换

Laplace 变换和反变换的定义为

$$F(s)=\int_0^{\infty} f(t)\mathrm{e}^{-st}\mathrm{d}t \tag{2.5-3}$$

$$f(t)=\frac{1}{2\pi j}\int_{c-j\infty}^{c+j\infty} F(s)\mathrm{e}^{st}\mathrm{d}s \tag{2.5-4}$$

与 Fourier 变换的机器实现相似，Laplace 变换与反变换的机器实现也有两条途径：直接调用指令 laplace 和 ilaplace；根据式(2.5-3)和式(2.5-4)定义，利用积分指令 int 实现。比

较而言，直接使用 laplace 和 ilaplace 指令实现机器变换显得较为简洁。

Fs=laplace(ft,t,s) 求“时域”函数 ft 的 Laplace 变换 Fs

ft=ilaplace(Fs,s,t) 求“频域”函数 Fs 的 Laplace 反变换 ft

说明

- ft 是以 t 为自变量的“时域”函数；Fs 是以复频率 s 为自变量的“频域”函数。
- 在此给出的是 laplace ，ilaplace 指令的完整调用格式。虽然它们都有默认调用格式，但本书建议：用户应慎用或不用默认调用格式，以免出错。
- Laplace 变换由法国数学家、天文学家 Pierre-Simon，marquis de Laplace（1749—1827）发明并应用。

例【2.5-4】 分别求 $e^{-at}\sin bt$，$u(t-a)$，$\delta(t-b)$，t^n 的 Laplace 变换。本例演示：符号参数的属性限定；dirac，heaviside 的调用。

（1）Laplace 变换指令的简单运用

```
syms t s a b
f1 = exp( - a * t) * sin(b * t)     % 据变换定义知，被变换函数不必乘 Heavivide 函数     <2>
F1 = laplace(f1,t,s)
f1 =
sin(b * t)/exp(a * t)
F1 =
b/((a + s)^2 + b^2)
```

（2）$u(t-a)$中参数 a 对 Laplace 变换的影响

```
sym a clear                         % 清除对 a 的任何限制性假设                          <4>
f2 = heaviside(t - a)
F2 = laplace(f2,t,s)                % 由于 a 的正负不确定，使得变换失败
ans =
a
f2 =
heaviside(t - a)
F2 =
laplace(heaviside(t - a), t, s)

syms a positive                     % 把 a 限制为正数                                    <7>
F3 = laplace(f2)                    % 变换成功
F3 =
1/(s * exp(a * s))
```

（3）$\delta(t-b)$中参数 b 对 Laplace 变换的影响

```
f4 = dirac(t - b);                  % 式中 b 的正负没有限定
F4 = laplace(f4,t,s)                % 给出两种可能结果
F4 =
piecewise([b < 0, 0], [0 <= b, 1/exp(b * s)])

f5 = dirac(t - a);                  % 式中 a 是被限定为正数的                            <11>
```

```
F5 = laplace(f5,t,s)              % 给出正确结果
ft_F5 = ilaplace(F5,s,t)          % 反变换验算
F5 =
1/exp(a * s)
ft_F5 =
dirac(a - t)
```

(4) t^n 参数 n 对 Laplace 变换的影响

```
n = sym('n','clear');             % 定义不带任何限制性假设的符号变量 n          <14>
F6 = laplace(t^n,t,s)             % 给出分段函数结果
F6 =
piecewise([ - 1 < Re(n), gamma(n + 1)/s^(n + 1)])

n = sym('n','positive')           % 把 n 定义为“正数”符号变量                  <16>
F6 = laplace(t^n,t,s)             % 得到正确变换结果                            <17>
n =
n
F6 =
gamma(n + 1)/s^(n + 1)
```

说明

- 由于 Laplace 变换是对 $t \geqslant 0$ 定义的函数实施的，所以被变换函数不必再专门用 Heviside 进行“因果处理”。(参见指令〈2〉)
- 本例中指令〈7〉〈16〉是对符号变量进行限制性假设的两种不同方式。
- 要清除已经对符号变量所做的限制性假设，必须采用如指令〈4〉或〈14〉那样的格式进行。关于符号变量限制性假设的清除更详细的说明，请见第 2.1.5 - 3 小节。
- 指令〈11〉定义的函数是连续域里的单位脉冲函数，也称 Dirac delta 函数。它的一种比较浅显而常见的定义是

$$\delta(t) = \begin{cases} 0 & t \neq 0 \\ \infty & t = 0 \end{cases}$$

$$\int_{-\infty}^{\infty} \delta(t)\mathrm{d}t = 1$$

Dirac delta 函数由英国理论物理学家 Paul Adrien Maurice Dirac (1902 — 1984) 首先提出并应用。

- 关于 Gamma 函数 $\Gamma(a) = \int_0^{\infty} \mathrm{e}^{-t} t^{a-1} \mathrm{d}t$ 的说明：当 a 为正整数($n+1$)时，下式成立。

$$\Gamma(n+1) = n! = \prod_{k=1}^{n} k$$

2.5.3　Z 变换及其反变换

一个离散因果序列的 Z 变换及其反变换的定义为

$$F(z) = \sum_{n=0}^{\infty} f(n) z^{-n} \tag{2.5-5}$$

$$f(n) = Z^{-1}\{F(z)\} \tag{2.5-6}$$

涉及Z反变换具体计算的方法，最常见的有三种：幂级数展开法、部分分式展开法和围线积分法。MATLAB的Symbolic Toolbox采用围线积分法设计了求取Z反变换的iztrans指令，相应的数学表达式是 $f(n)=\dfrac{1}{2\pi j}\oint_{\Gamma}F(z)z^{n-1}\mathrm{d}z$。

FZ=ztrans(fn,n,z)　　求"时域"序列fn的Z变换FZ

fn=iztrans(FZ,z,n)　　求"频域"序列FZ的Z反变换fn

说明

- 这两个指令的调用格式及注意事项，与前两节指令情况相同。
- Z变换最先由Zadeh和Ragazzini于1952年提出。Lotfali Askar－Zadeh，美国数学家、计算机科学家(1921－)，模糊数学创始人。

例【2.5-5】 一组Z变换、反变换算例。本例演示：ztrans及iztrans的使用方法；变量的限定性假设；限定性假设的解除；Kronecker Delta离散单位脉冲函数；离散单位阶跃函数；evalin指令的应用。

(1) $6\left(1-\left(\dfrac{1}{2}\right)^n\right)$序列的Z变换。简单的指令应用

```
clear
syms n z clear                          % 删除关于 n,z 的限定性假设              <2>
gn = 6 * (1 - (1/2)^n)                  % 定义时间序列
G = simple(ztrans(gn,n,z));             % 实施变换,并简化
pretty(G)                               % 易读形式显示
gn =
6 - 6 * (1/2)^n

       6 z
  -------------
     2
  2 z  - 3 z + 1
```

(2) 采样周期为 T 的 $\sin(\omega\cdot nT)$序列的Z变换及反变换

```
syms n w T z clear                      % 删除限定性假设                         <6>
fwn = sin(w * n * T);                   % 定义时间序列
FW = ztrans(fwn,n,z);                   % Z 变换
pretty(FW),disp(' ')
inv_FW = iztrans(FW,z,n)                % 实施 Z 反变换

       z sin(T w)
  ----------------------
   2
  z  - 2 cos(T w) z + 1

inv_FW =
sin(T * n * w)
```

(3) 单位阶跃序列 $f(n)=1$ 的 Z 变换及反变换

```
syms n z clear                                      % 删除限定性假设                <11>
f1 = 1;
F1 = ztrans(f1,n,z);
pretty(F1)
inv_F1 = iztrans(F1,z,n)

    z
  -----
  z - 1
inv_F1 =
1
```

(4) 单位脉冲序列 $\delta(n)=\begin{cases}1 & n=0\\0 & else\end{cases}$ 的 Z 变换及反变换

```
clear
syms n z clear                                      % 删除限定性假设                <17>
delta = sym('kroneckerDelta(n, 0)');                % 定义单位脉冲                  <18>
KD = ztrans(delta,n,z)
inv_KD = iztrans(KD)
KD =
1
inv_KD =
kroneckerDelta(n, 0)
```

(5) $f(n)\delta(n-k)$序列的 Z 变换及反变换

```
syms n z clear                                      % 删除限定性假设                <22>
k = sym('k','positive');                            % 对 k 最多只能做"正数"假设     <23>
fd = sym('f(n) * kroneckerDelta(n - k, 0)');
FD = ztrans(fd,n,z)
inv_FD = iztrans(FD,z,n)
FD =
piecewise([k in Z_, f(k)/z^k], [Otherwise, 0])
inv_FD =
piecewise([k in Z_, f(k) * kroneckerDelta(k - n, 0)], [Otherwise, 0])
```

(6) Z 表达式 $g(z)=e^{-a/z}$的反变换

```
syms a z n clear
GZ = exp( - a/z);                                   % 定义表达式
gn = iztrans(GZ,z,n)                                % 进行反变换                    <29>
gn =
( - a)^n/factorial(n)
```

说明

- 指令〈2〉〈6〉〈11〉〈17〉〈22〉清除限定性假设。这些指令的设置，使为了防止读者因操作各段程序的次序不同而引出错误结果。

- kroneckerDelta(n－k, 0)是 MuPAD 定义的单位脉冲。其数学表达式为

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

对于 kroneckerDelta(n－k, 0)而言，仅当 n－k=0 时，函数值为 1，其他地方均为 0。

- Kronecher Delta 函数由德国数学家、逻辑学家 Leopold Kronecker (1823－1891) 提出并应用。
- 指令〈29〉运行结果中的 factorial(n)是 MuPAD 的定义函数“n 的阶乘”，其数学表达为

$$n! = \begin{cases} 1 & n = 0 \\ n(n-1)! & n > 0 \end{cases}$$

- 顺便介绍如何获得关于 kroneckerDelta，factorial 等的信息。在 MATLAB 指令窗中运行 doc(symengine,'delta')就能引出 MuPAD 的帮助浏览窗，并从中找到 kroneckerDelta 的相关解释。寻找 factorial 的方法类似。

2.5.4 符号卷积

由于卷积在信号处理和系统动态特性研究中占有特殊的地位，本书在此专辟一节以算例方式讨论符号卷积的机器实现。

例【2.5-6】 已知系统冲激响应 $h(t)=\frac{1}{T}e^{-t/T}U(t)$，求 $u(t)=e^{-t}U(t)$输入下的输出响应。本例演示：卷积的时域积分法；simple 的反复简化。

由系统分析可知，输出响应等于卷积 $y(t)=\int_0^t u(\tau)h(t-\tau)d\tau$。据此，可推出该题的理论计算结果是 $y(t)=\frac{1}{T-1}(e^{-\frac{t}{T}}-e^{-t})U(t)$。下面是计算机实现的指令。

```
syms T t tao
ut = exp( - t);                                   %定义系统输入
ht = exp( - t/T)/T;                               %定义系统冲激响应
uh_tao = subs(ut,t,tao) * subs(ht,t,t - tao);     %运用变量替换指令形成被积函数
yt = simple(simple(int(uh_tao,tao,0,t)))          %实施卷积
yt =
-(1/exp(t) - 1/exp(t/T))/(T - 1)
```

例【2.5-7】 采用 Laplace 变换和反变换求上例的输出响应。本例演示：通过变换法求卷积，即系统冲激响应。

对式 $y(t)=\int_0^t u(\tau)h(t-\tau)d\tau$ 两边进行 Laplace 变换得 $L[y(t)]=L[u(t)]\cdot L[h(t)]$，因此有 $y(t)=L^{-1}\{L[y(t)]\}=L^{-1}\{L[u(t)]\cdot L[h(t)]\}$。于是用以下指令可得结果。

```
syms s
yt = ilaplace(laplace(ut,t,s) * laplace(ht,t,s),s,t);
yt = simple(yt)
yt =
-(1/exp(t) - 1/exp(t/T))/(T - 1)
```

2.6　符号矩阵分析和代数方程解

2.6.1　符号矩阵分析

最常用的矩阵分析和解算指令如表 2.6－1 所示。

表 2.6－1　符号矩阵分析和解算指令汇总表

指　令	含　义	参考节次
colspace(A)	矩阵的列空间基	
det(A)	行列式 $\|\boldsymbol{A}\|$	例 2.6－1
diag(A)	取对角元构成向量，或据向量构成对角阵	
[V, D]=eig(A)	特征值分解，使 $\boldsymbol{AV}=\boldsymbol{VD}$	例 2.6－1
expm(A)	矩阵指数 e^A	
inv(A)	矩阵逆 $\boldsymbol{A}^{-1}$	例 2.6－1
[V, J]=jordan(A)	准确阵 $\boldsymbol{A}$ 的 *Jordan* 分解，使 $\boldsymbol{AV}=\boldsymbol{VJ}$	
null(A)	零空间的基	
poly(A)	矩阵的特征多项式	
rank(A)	矩阵秩	
rref(A)	$\boldsymbol{A}$ 的行阶梯形式	
s=svd(A)[U,S,V]=svd(vpa(A))	奇异值分解	
tril(A)	$\boldsymbol{A}$ 的下三角形式	
triu(A)	$\boldsymbol{A}$ 的上三角形式	

说明

- 以下矩阵比较适于进行符号分析和计算：
  - 规模很小的非数字矩阵；
  - 元素是整数或小数字有理分数构成的矩阵；
  - 借助 *vpa* 指令产生的任意精度矩阵；
  - 高等学校教科书上的矩阵。
- *Jordan* 分解对矩阵元素及其敏感，因此建议：需进行 *Jordan* 分解的矩阵应采用准确数字构成。

例【2.6－1】 求矩阵 $\boldsymbol{A}=\begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix}$ 的行列式、逆和特征根。本例演示：符号矩阵的输入方式；符号矩阵运算结果的准确性和繁琐。

（1）很小规模的参数矩阵的符号分析

```
syms a11 a12 a21 a22
A = [a11,a12;a21,a22]
```

```
DA = det(A)
IA = inv(A)

A =
[ a11, a12]
[ a21, a22]
DA =
a11 * a22 - a12 * a21
IA =
[  a22/(a11 * a22 - a12 * a21), - a12/(a11 * a22 - a12 * a21)]
[ - a21/(a11 * a22 - a12 * a21),  a11/(a11 * a22 - a12 * a21)]
```

(2) 借助公因子表达的参数矩阵特征值分析

```
EA = subexpr(eig(A),'D')
D =
(a11^2 - 2 * a11 * a22 + a22^2 + 4 * a12 * a21)^(1/2)
EA =
 a11/2 + a22/2 - D/2
 a11/2 + a22/2 + D/2
```

例【2.6-2】 著名的 Givens 旋转(变换)$G=\begin{bmatrix}\cos t & -\sin t\\ \sin t & \cos t\end{bmatrix}$对矩阵 $A=\begin{bmatrix}\sqrt{3}/2 & 1/2\\ 1/2 & \sqrt{3}/2\end{bmatrix}$的旋转作用。本例演示:Givens 旋转的几何意义;符号矩阵乘法;符号矩阵的数值化;绘图指令的配合。

(1) 产生符号矩阵,和旋转变换

```
syms t
A = sym([sqrt(3)/2,1/2;1/2,sqrt(3)/2])
G = [cos(t), - sin(t);sin(t),cos(t)];                % 一般的 Givens 矩阵
GA = G * A
A =
[ 3^(1/2)/2,         1/2]
[         1/2, 3^(1/2)/2]
GA =
[ (3^(1/2) * cos(t))/2 - sin(t)/2, cos(t)/2 - (3^(1/2) * sin(t))/2]
[ cos(t)/2 + (3^(1/2) * sin(t))/2, sin(t)/2 + (3^(1/2) * cos(t))/2]
```

(2) 图示 Givens 旋转(取转角 t=110 度)

```
clf
An = subs(GA,t,110/180 * pi);                 % 转角为 110 度的 Givens 数值矩阵
Op = [0;0];                                   % 原点坐标向量,为绘图用。
Ad = double(A);                               % 符号矩阵数值化
v1 = [Op,Ad(:,1)]';v2 = [Op,Ad(:,2)]';        % 形成 A 矩阵的列向量
u1 = [Op,An(:,1)]';u2 = [Op,An(:,2)]';        % 构造经旋转后的 GA 矩阵的列向量
plot(v1(:,1),v1(:,2),'--k',v2(:,1),v2(:,2),'b')
axis([-1,1,-1,1]),axis square
hold on
```

```
hu=plot(u1(:,1),u1(:,2),'--k',u2(:,1),u2(:,2),'b');
set(hu,'LineWidth',4)
title('Givens Rotation')
Lstr=['旋转前的 v1';'旋转前的 v2';'旋转后的 u1';'旋转后的 u2'];   %        <16>
legend(Lstr,'Location','South')                                 %        <17>
hold off
grid on
```

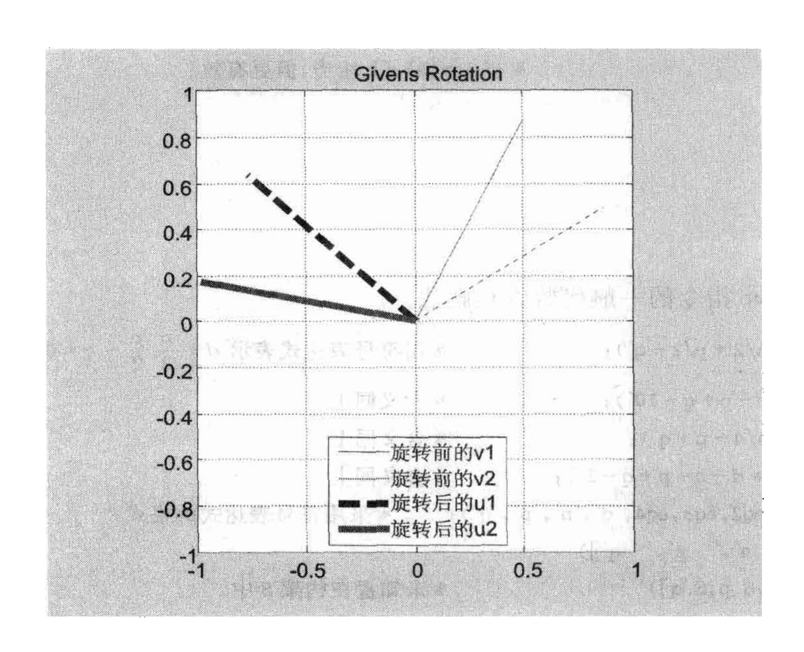


图 2.6-1 Givens 旋转的几何意义

说明

注意指令〈16〉中各字符串间用“分号”分隔。

2.6.2 线性方程组的符号解

矩阵计算是求解线性方程组最简便有效的方法。在 MATLAB 中，不管数据对象是数值还是符号，实现矩阵运算的指令形式几乎完全相同。因此，关于求解线性方程组符号解的问题，读者可套用求数值解的方法进行，请参见第 4.2 节，不冗述。

例【2.6-3】 求线性方程组

$$\begin{cases} d+\frac{n}{2}+\frac{p}{2}=q \\ n+d+q-p=10 \\ q+d-\frac{n}{4}=p \\ q+p-n-8d=1 \end{cases}$$

的解。本例演示：符号线性方程组的矩阵除求解法；solve 求代数方程解。

（1）采用矩阵除的线性方程解法

该方程组的矩阵形式是 $\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{2} & -1 \\ 1 & -\frac{1}{4} & -1 & 1 \\ -8 & -1 & 1 & 1 \end{bmatrix}\begin{bmatrix} d \\ n \\ p \\ q \end{bmatrix}=\begin{bmatrix} 0 \\ 10 \\ 0 \\ 1 \end{bmatrix}$。该式简记为 $\boldsymbol{AX}=\boldsymbol{b}$。

求符号解的指令如下

```
A = sym([1 1/2 1/2 -1;1 1 -1 1;1 -1/4 -1 1;-8 -1 1 1]);
b = sym([0;10;0;1]);
X1 = A\b                              % 与 inv(A) * b 相当,但更有效。
X1 =
 1
 8
 8
 9
```

（2）采用 solve 指令的一般代数方程解法

```
eq1 = sym('d + n/2 + p/2 - q');              % 用符号表达式表示 d + n/2 + p/2 - q = 0      <4>
eq2 = sym('d + n - p + q - 10');             % 含义同上
eq3 = sym('d - n/4 - p + q');                % 含义同上
eq4 = sym(' - 8 * d - n + p + q - 1');       % 含义同上
S = solve(eq1,eq2,eq3,eq4,'d','n','p','q');      % 采用符号表达式的格式      <8>
disp(['  d','  n','  p','  q'])
disp([S.d,S.n,S.p,S.q])                      % 未知量在构架 S 中      <10>
  d  n  p  q
[ 1, 8, 8, 9]
```

说明

- 在 MATLAB 中，不管是符号矩阵，还是数值矩阵，都用"+，-，*，/"定义了矩阵的加、减、乘、除。
- solve 是解线性或非线性代数方程的通用指令。详细说明见下节。

2.6.3 一般代数方程组的解

这里所讲的一般代数方程包括线性（Linear）、非线性（Nonlinear）和超越方程（Transcedental equation）等，求解指令是 solve。当方程组不存在符号解时，若又无其他自由参数，则 solve 将给出数值解。该指令最清晰的使用格式如下。

```
S=solve('eq1', 'eq2', …, 'eqn', 'v1', 'v2',…, 'vn')      求方程组关于指定变量的解（推荐）
S=solve(exp1, exp2, …, expn, v1, v2, …, vn)              求方程组关于指定变量的解（可用）
```

说明

- 'eq1'，'eq2'，…，'eqn' 或是字符串表达的方程，或是字符串表达式；'v1'，'v2'，…，'vn' 是字符串表达的求解变量名。
- exp1，exp2，…，expn 只能是符号表达式；v1，v2，…，vn 是求解的符号变量。

- 如 eq1，eq2，…，eqn 是不含“等号”的表达式，则指令认定为是对 eq1=0，eq2=0，…，eqn=0 求解。
- S 是一个构架数组。如要显示求解结果，必须采用 S. v1，S. v2，…，S. vn 的援引方式。关于构架，请参见本书附录 A. 3。指令 solve 在默认规则下，还有一些形式更为简单的调用方式。但本书认为，只有当读者对方程组默认变量及其次序充分理解时，才适于使用。否则容易引起混乱。那时，变量次序是按 findsym 所执行的规律认定的。
- 在得不到“封闭型解析解”时，如果又不存在其他不确定参数，那么给出数值解。

例【2.6-4】 求方程组

$$\begin{cases} uy\hat{}2 + vz + w = 0 \\ y + z + w = 0 \end{cases}$$

关于 y,z 的解。本例演示：指令调用格式的正确使用。

```
S = solve('u * y^2 + v * z + w = 0','y + z + w = 0','y','z')          % 采用字符串方程的格式          <1>
disp('S.y'),disp(S.y),disp('S.z'),disp(S.z)                           % 未知量在 S 构架中             <2>
S =
    y: [2x1 sym]
    z: [2x1 sym]
S.y
 (v + 2*u*w + (v^2 + 4*u*w*v - 4*u*w)^(1/2))/(2*u) - w
 (v + 2*u*w - (v^2 + 4*u*w*v - 4*u*w)^(1/2))/(2*u) - w
S.z
 -(v + 2*u*w + (v^2 + 4*u*w*v - 4*u*w)^(1/2))/(2*u)
 -(v + 2*u*w - (v^2 + 4*u*w*v - 4*u*w)^(1/2))/(2*u)
```

说明

- 以下格式的指令都能给出关于 y，z 的正确解。

```
S=solve('u * y^2+v * z+w=0','y+z+w=0','z', 'y')           在此，指定变量次序没影响
S=solve('u * y^2+v * z+w','y+z+w','z,y')                  注意：'z,y'中，y 前不能有空格
syms y z u v w , S=solve(u * y^2+v * z+w, y+z+w, y, z)
[y,z]=solve('u * y^2+v * z+w=0','y+z+w=0','y','z')        输出宗量次序正确，解才正确
```

- 由于没指定变量，findsym 把 w，y 依次认作变量。所以，以下格式的指令不能给出关于 y，z 的正确解。

```
[y,z]=solve('u * y^2+v * z+w=0','y+z+w=0')                实际上给出关于 w，y 的解
S=solve('u * y^2+v * z+w=0','y+z+w=0')                    给出关于 w，y 的解 S.w，S.y
```

  在此，S 是构架，w 和 y 分别是构架 S 的域。（关于构架更详细内容请参看附录 A. 3。）
- 以下指令求解时，总是按字母排列解的次序。因此一旦指令中输出宗量名的字母次序混乱，结果就错。

```
[z,y]=solve('u * y^2+v * z+w=0','y+z+w=0','y,z')          由于 y，z 分赋于 z，y，造成混乱
[z,y]=solve('u * y^2+v * z+w=0','y+z+w=0','z','y')        由于 y，z 分赋于 z，y，造成混乱
```

- 以下格式中包含的是符号方程，而不是符号表达式，因此无法运行。

```
syms y z u v w , S=solve(u * y^2+v * z+w=0 , y+z+w=0 , y , z)
```



例【2.6-5】 solve 指令求

$$\begin{cases} d+\dfrac{n}{2}+\dfrac{p}{2}=q \\ n+d+q-p=10 \\ q+d-\dfrac{n}{4}=p \end{cases}$$

构成的“欠定”方程组解。本例演示：solve 指令的输出格式；含自由变量的欠定方程解。

```
syms d n p q
eq1 = d + n/2 + p/2 - q;eq2 = n + d + q - p - 10;eq3 = q + d - n/4 - p;     % <2>
S = solve(eq1,eq2,eq3,d,n,p,q);                                              % <3>
disp(['    S.d','    S.n','    S.p','        S.q'])
disp([S.d,S.n,S.p,S.q])
   S.d     S.n    S.p         S.q
[ z/3 - 2, 8, (4 * z)/3 - 4, z]
```

说明

注意：当方程数少于独立变量数，所以解中一定含有不定参数。在本例中所给的解中，z 表示不定参数。

例【2.6-6】 求$(x+2)^x=2$的解。本例演示：无解析解时，给出近似解。

```
clear all,syms x;
s = solve('(x + 2)^x = 2','x')
xs = (s(1) + 2)^s(1)                     % 验算                    <3>
s =
matrix([[0.69829942170241042826920133106081]])
xs =
2.0
```

说明

- 本例中，由 solve 解得的 s 是个 MuPAD 格式的矩阵，尽管只有一个元素。按 MuPAD 规则，矩阵不能与标量相加，而且“矩阵的指数矩阵”没有定义。所以，假如在验算时，采用(s+2)^s 的格式运作，那会出错。
- s(1)是矩阵 s 的元素，是标量，因此，验算式采用形如指令〈3〉的格式。

2.7 代数状态方程求符号传递函数

在“信号和系统”或“自动控制原理”教科书中，几乎都有专门章节用于讲授“梅逊算法”和“结构框图算法”。实现这些算法依靠的是一些特定“规则”或“技巧”，完成这些算法需要的只是“一张纸”和“一支笔”。

利用信号流图（Signal - flow graphs）求解系统传递函数的原始思想由 S. J. Mason（1921—1974）在 1953 年提出。经其对代数方程 Cramer 求解法的长期深入研究，于 1960 年归纳成著名的梅逊增益公式（Mason gain formula）

$$\frac{Y}{U}=\frac{1}{\Delta}\cdot\sum_{k=1}^{n}p_k\Delta_k$$

式中：Y、U 分别是信流图的输出、输入；Δ 是信流图的特征式（实际上即状态矩阵的行列式）；p_k 是从输入到输出的第 k 条前向通路增益；Δ_k 是与第 k 条前向通路对应的信流图余因式；n

是输入输出间的前向通路总数。

这两种产生于 20 世纪的 50、60 年代的“梅逊算法”和“结构框图算法”，一方面高度巧妙地把“高阶代数方程”求解问题转化为一系列“手工计算规则”，为控制系统建模、电路分析和设计作出了卓越的贡献；另一方面，它们的产生，在一定意义上，也是出于“缺乏高阶线性方程解算工具”的历史无奈。

时至今日，虽然文献积淀、知识传承、历史惯性和算法本身的“手工可算性”，使得现在使用的相当一些教科书、工程文件和科研文献中仍包含“结构框图算法”和“梅逊算法”这方面的内容。但面对愈益复杂的系统，特别是多变量系统，这些手工算法暴露出了过于繁琐、技巧性高、特别费神的缺陷。使用者稍不留神，就得不到正确结果。

系统（符号）传递函数求取问题的原始本质是“符号代数方程组的求解问题”。基于对这原始本质的认识，本节将集中描述求取（符号）传递函数的“代数状态方程法”。在此，冠以“代数”修饰词，是为了区别于那源自“微分方程”或“差分方程”的状态方程。但该方法无论从形式上还是本质上，都与 S－传递函数求取的“微分状态方程法”、Z－传递函数求取的“差分状态方程法”十分相似。

为节省篇幅和易于读者理解，“代数状态方程法”求取符号传递函数的步骤和 M 码实现分两小节以不同算例进行。

2.7.1　结构框图的代数状态方程解法

例【2.7－1】 求图 2.7－1 所示某三环系统的传递函数。本例演示：(A)系统“代数状态方程”的建立；(B)根据代数状态方程求系统的传递函数；(C)编写 M 码时，系统矩阵的输入采用“全元素赋值法”；(D)sort 指令对符号表达式的排序功能。

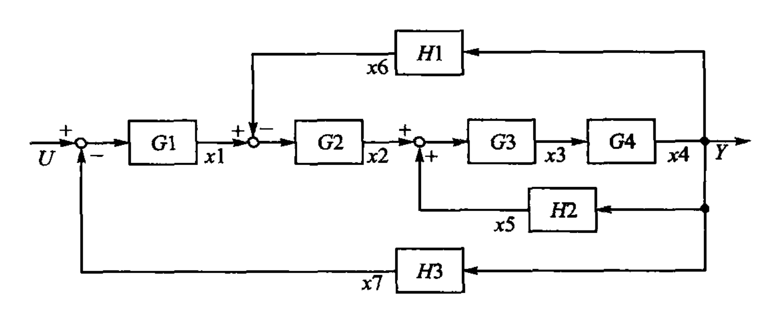


图 2.7－1　三环系统的结构框图

(1) 在结构框图上标识状态变量

参照箭头流向，把结构框图中各方块的输出量依次标识为状态变量 $x_1, x_2, \cdots, x_7$。

(2) 建立代数状态方程

$$\begin{cases} \boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}U \\ Y = \boldsymbol{c}\boldsymbol{x} \end{cases} \tag{2.7-1}$$

根据结构框图，填写式(2.7－1)中矩阵 $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}$ 的各元素。即把式(2.7－1)具体化为

$$\left\{\begin{aligned}
&\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -G_1 \\ G_2 & 0 & 0 & 0 & 0 & -G_2 & 0 \\ 0 & G_3 & 0 & 0 & G_3 & 0 & 0 \\ 0 & 0 & G_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & H_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & H_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & H_3 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} G_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot U \\
&Y = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}
\end{aligned}\right. \tag{2.7-2}$$

(3) 据代数状态方程求传递函数的理论演绎

对式(2.7-1)的第一个方程进行整理，可以写出

$$\boldsymbol{x} = (\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{b}U \tag{2.7-3}$$

再把此式代入式(2.7-1)的第二个方程，即输出方程，可得

$$Y = \boldsymbol{c}(\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{b}U$$

进而可得传递函数

$$G = \frac{Y}{U} = \boldsymbol{c}(\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{b} \tag{2.7-4}$$

(4) 代数状态方程法计算传递函数的 M 码

当系统矩阵 $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}$ 的规模较小时，采用“全元素赋值法”进行编码也许是直观和适当的。具体如下：

```
syms G1 G2 G3 G4 H1 H2 H3
A = [   0,    0,    0,    0,    0,    0, - G1;
       G2,    0,    0,    0,    0, - G2,    0;
        0,   G3,    0,    0,   G3,    0,    0;
        0,    0,   G4,    0,    0,    0,    0;
        0,    0,    0,   H2,    0,    0,    0;
        0,    0,    0,   H1,    0,    0,    0;
        0,    0,    0,   H3,    0,    0,    0];
b = [  G1;    0;    0;    0;    0;    0;    0];
c = [   0,    0,    0,    1,    0,    0,    0];
Y2Ua = c * ((eye(size(A))- A)\b);        % 利用“左除”取代“求逆”，计算传递函数
[NN,D0] = numden(Y2Ua);D0                % 分离出分子、分母多项式                    <12>
DD = sort(D0)                            % 分母多项式排序                            <13>
disp([blanks(5),'传递函数 Y2Ua 为'])
pretty(NN/DD)
```

```
DO =
G3 * G2 * G1 * H3 * G4 + G3 * G2 * H1 * G4 + 1 - G3 * H2 * G4

传递函数 Y2Ua 为
                              G3 G4 G2 G1
              -------------------------------------------------------
              G1 G2 H3 G4 G3  +  G2 H1 G4 G3 - H2 G4 G3  +  1
DD =
G1 * G2 * G4 * H3 * G3 + G2 * G4 * G3 * H1 - G4 * H2 * G3 + 1
```

说明

- 本例所演示的算法，对更为复杂的结构框图也适用。换句话说，代数状态方程的建立方法、根据状态方程求取传递函数的"程式"、具体算法的 M 码等，都具有通用性。
- 图 2.7-1 所示结构框图是许多"自动控制原理"及"信号和系统"教科书中的典型例题。有兴趣的读者可以进行比较对照。
- 在编写程序时，(10×10)以下规模矩阵的输入，采用"全元素赋值法"也许是适当的，因为这种输入法比较直观。

2.7.2 信号流图的代数状态方程解法

例【2.7-2】 作为比较，画出图 2.7-1 所示 结构框图的等价信号流图，并据此信号流图运用"代数状态方程"求系统的传递函数。本例演示：(A)信号流图的代数状态方程的建立；(B)根据代数状态方程求传递函数；(C)在 $G_1=\frac{100}{s+10}$，$G_2=\frac{1}{s+1}$，$G_3=\frac{s+1}{s^2+4s+4}$，$G_4=\frac{s+1}{s+6}$，$H_1=\frac{2s+12}{s+1}$，$H_2=\frac{s+1}{s+2}$，$H_3=1$ 的情况下，求取参数具体化的传递函数。

(1) 根据图 2.7-1 画出相应的信号流图

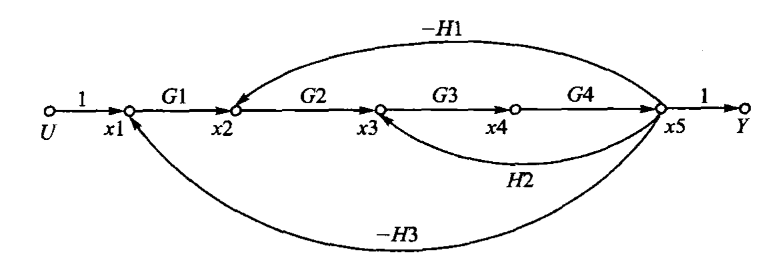


图 2.7-2 三环系统的信号流图

(2) 代数状态方程法求取信号流图传递函数的数学原理

首先对信号流图的节点进行如图 2.7-2 那样的状态变量 $x_1, x_2, \cdots, x_5$ 标识，然后根据信号流向写出如下状态代数方程。

$$\begin{cases}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\end{bmatrix}=\begin{bmatrix}0&0&0&0&-H_3\\G_1&0&0&0&-H_1\\0&G_2&0&0&H_2\\0&0&G_3&0&0\\0&0&0&G_4&0\end{bmatrix}\cdot\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\end{bmatrix}+\begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix}\cdot U\\ Y=[0\quad 0\quad 0\quad 0\quad 1]\cdot\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\end{bmatrix}\end{cases}\tag{2.7-5}$$

此式可简记为

$$\begin{cases}\boldsymbol{x}=\boldsymbol{Ax}+\boldsymbol{b}U\\Y=\boldsymbol{cx}\end{cases}$$

并据此写出传递函数

$$G=\frac{Y}{U}=\boldsymbol{c}(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{b}\tag{2.7-6}$$

(2) 实现以上算法的 M 码

```
syms G1 G2 G3 G4 H1 H2 H3
A=[    0,   0,   0,   0,   -H3;
      G1,   0,   0,   0,   -H1;
       0,  G2,   0,   0,    H2;
       0,   0,  G3,   0,     0;
       0,   0,   0,  G4,     0];
b=[    1;   0;   0;   0;     0];
    c=[0,   0,   0,   0,     1];
Y2Ub=c*((eye(size(A))-A)\b);         %求传递函数
disp([blanks(5),'传递函数 Y2Ub 为'])
pretty(Y2Ub)
```

```
    传递函数 Y2Ub 为
                           G1 G2 G3 G4
      ------------------------------------------------
      G2 G3 G4 H1 - G3 G4 H2 + G1 G2 G3 G4 H3 + 1
```

(4) 方块参数具体化时的传递函数

```
syms s
Sblock={100/(s+10),1/(s+1),(s+1)/(s^2+4*s+4),(s+1)/(s+6),(2*s+12)/(s+1),
(s+1)/(s+2),1};                                   %模块的传递函数          <9>
ww=subs(Y2Ub,{G1,G2,G3,G4,H1,H2,H3},Sblock);    %传递函数具体化替代        <10>
Y2Uc=simple(ww);
[NN,DD]=numden(Y2Uc);                             %分离出分子、分母多项式
NN=expand(NN);                                    %分子多项式展开
disp('参数具体化的传递函数 Y2Uc 为')
```

```
pretty(NN/DD)
参数具体化的传递函数 Y2Uc 为

                   2
          100 s    + 300 s + 200
-------------------------------------------------
 5        4         3         2
s   + 21 s   + 157 s   + 663 s   + 1301 s + 910
```

说明

- 把本例计算结果 Y2Ub 与上例的计算结果 Y2Ua 进行比较，显然，两者完全相同。
- 本例第〈10〉行指令实施变量置换。请注意：指令中“花括号”的用法。
- 本例所得的“参数具体化传递函数 Y2Uc”与例 7.1－2 结果相同。
- 本例方法可以推广应用于复杂的多输入多输出系统。

2.8 符号计算结果的可视化

符号计算结果的可视化有两条途径：一，利用计算获得的符号表达式直接绘图；二，据获得的符号表达式或符号数值结果，进而转换得到数值数据，再利用 MATLAB 的数值绘图指令绘制所需的图形。

2.8.1 直接可视化符号表达式

MALTAB 中有一组专门实现函数可视化的指令，它们的名称特点是：名称前两个字符都是 ez，其含义为 Easy to。表 2.8－1 列出了这些指令的名称。这组指令适用于多种类型函数：符号函数、字符串函数、M 文件函数和句柄函数等。

表 2.8－1　简捷作图指令汇总

指令名	含　义	可执行示例
ezcontour	画等位线	ezcontour('cos(x+sin(y))－sin(y)'),colormap(jet)
ezcontourf	画填色等位线	colormap(flipud(cool)),ezmesh('sin(x) * sin(y)') hidden off,hold on, ezcontourf('sin(x) * sin(y)'),view([34,62]) ,hold off
ezmesh	画网线图	ezmesh('exp(－s) * cos(t)','exp(－s) * sin(t)','t',[0,8,0,4 * pi])
ezmeshc	画带等位线的网线图	ezmeshc('y/(1 + x^2 + y^2)',[－5,5,－2 * pi,2 * pi])
ezplot	画二维曲线	ezplot('1/y－log(y)+log(－1+y)+x － 1')
ezplot3	画三维曲线	ezplot3('sin(3 * t) * cos(t)','sin(3 * t) * sin(t)','t','animate')
ezpolar	画极坐标曲线	ezpolar('sin(tan(t))')
ezsurf	画曲面图	ezsurf('(x+8) * ((y)^2)/((x+8)^2 + (y)^4+eps)','circ') shading interp,colormap(flipud(hot)),view([83,84])
ezsurfc	画带等位线的曲面图	ezsurfc('sin(x) * sin(y)')

1. 单独立变量符号函数的可视化

ezplot(Fx,[xmin,xmax,ymin,ymax])	在指定 x 和 y 范围内，绘制 $y=f(x)$ 描写的平面曲线
ezplot(Fxy,[xmin,xmax,ymin,ymax])	在指定 x 和 y 范围内，绘制 $f(x,y)=0$ 描写的平面曲线
ezplot(xt,yt,[tmin,tmax])	在指定 t 范围内，绘制 $x=x(t)$，$y=y(t)$ 描写的平面曲线
ezplot3(xt,yt,zt,[tmin,tmax])	在指定 t 范围内，绘制 $[x(t),y(t),z(t)]$ 描写的三维空间曲线

说明

- 平面曲线指令 ezplot 的第一(和第二)输入量可以有三种形式：Fx；Fxy；xt,yt。它们分别表示的数学含义是：$y=f(x)$；$f(x,y)=0$；$x=x(t)$，$y=y(t)$。但不管何种表达方式，这些输入量描写的曲线一定有一个独立变量(见例 2.3-12、2.4-2、2.4-3、2.5-2 和 2.8-1)。
- 空间曲线指令 ezplot3 的前三个输入量 xt,yt,zt，都采用参数表达形式。注意：空间曲线也只有一个独立变量。
- 函数输入量 Fx;Fxy;xt,yt,zt 的程序表现可以是：符号函数、字符表达函数、函数 M 文件句柄、匿名函数句柄。
- ezplot 会自动把被绘函数和自变量分别标写为图名和横轴名。但用户也可以根据需要，使用 title,xlabel 指令重写图名(见例 2.8-1)和横轴名。
- ezplot 指令不能制定所绘曲线线型、色彩；不允许同时绘制多条曲线。但采取一些辅助措施，仍可实现色彩控制和重绘(见例 2.3-12)。
- text，text,grid,zoom,ginput 等指令可用于 ezplot 绘制的图形。

例【2.8-1】 绘制 $y=\frac{2}{3}e^{-\frac{t}{2}}\cos\frac{\sqrt{3}}{2}t$ 和它的积分 $s(t)=\int_0^t y(t)\mathrm{d}t$ 在 $[0,4\pi]$ 间的图形(见图 2.8-1)。本例演示：单一的“一元符号函数”的直接可视化；多子图图形窗；图名的自动标注和重写。

```
syms t tao
y = 2/3 * exp( - t/2) * cos(sqrt(3)/2 * t)          % 定义符号函数 y(t)
s = subs(int(y,t,0,tao),tao,t)                      % 获得积分函数 s(t)
subplot(2,1,1)                                      % 指定分上下子图中的“上子图”
ezplot(y,[0,4 * pi]),ylim([ - 0.2,0.7])
grid on                                             % 坐标纸上打网格
subplot(2,1,2)                                      % 指定“下子图”
ezplot(s,[0,4 * pi])
grid on
title('s = \int y(t)dt')                            % 重写下子图图名
y =
(2 * cos((3^(1/2) * t)/2))/(3 * exp(t/2))
s =
1/3 - (2 * cos((3^1/2) * t)/2)/2 - 3^(1/2) * sin((3^(1/2) * t)/2))/2))/(3 * exp(t/2))
```

说明

- 本例 ezplot 中的待画函数是“单一的符号表达式”，MATLAB 自动识别出变量 t 是“独

立自由变量”。体现 ezplot 这种用法的算例还有例 2.3-12、2.4-2、2.4-3、2.5-2 和例 2.8-1。

- 不管在二维平面上，还是在三维空间中，使用参变量画曲线时，每个“维度”变量都应表达为“单参变量”的函数。
- 函数绘图指令 ezplot 和数值绘图指令 plot 可以混合使用(见例 2.4-3)。

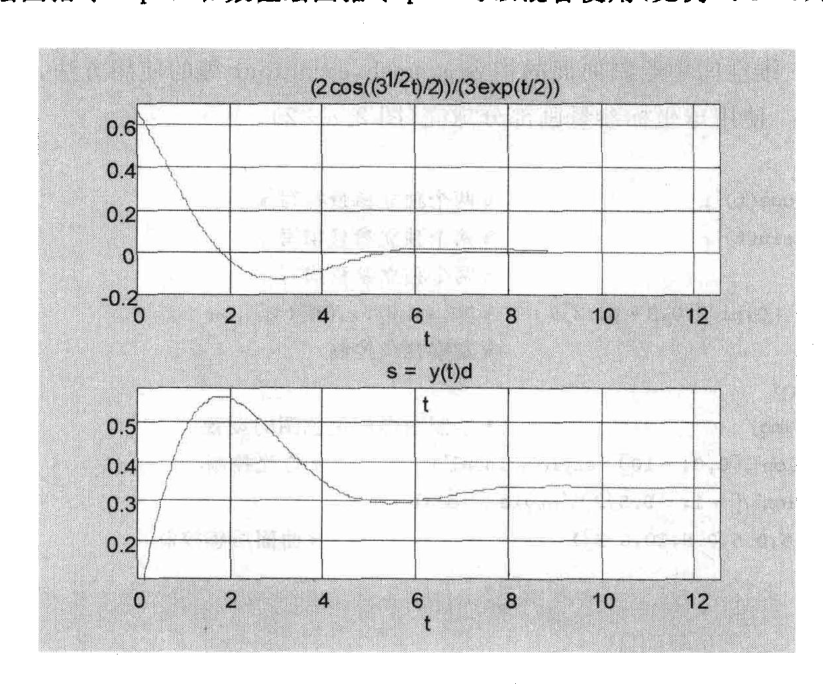


图 2.8-1 ezplot 使用示例

2. 双独立变量符号函数的可视化

ezsurf(Fxyz,dom_f)	在指定矩形域上画二元函数 $F(x,y,z)=0$ 曲面
ezsurf(Fxyz,dom_f,'circ')	在圆形域上画二元函数 $F(x,y,z)=0$ 曲面
ezsurf(x,y,z,dom_st,ngrid)	在指定矩形域上画 $x=x(s,t),y=y(s,t),z=z(s,t)$曲面
ezsurf(x,y,z,dom_st,'circ')	在指定圆形域上画 $x=x(s,t),y=y(s,t),z=z(s,t)$曲面

说明

- 曲面指令 ezsurf 的第一(和第二、第三)输入量可以有两种形式：Fxyz；x，y，z。它们分别表示的数学含义是：$F(x,y,z)=0$；$x=x(s,t),y=y(s,t),z=z(s,t)$。但不管何种表达方式，这些输入量描写的曲面一定有两个独立变量。
- 函数输入量 Fxyz；x，y，z 的程序表现可以是：符号函数、字符表达函数、函数 M 文件句柄、匿名函数句柄。
- dom_f 取二元数组[a, b] 时，自变量范围是 $a\leqslant x\leqslant b, a\leqslant y\leqslant b$；
  dom_f 取四元数组[a,b,c,d] 时，自变量范围是 $a\leqslant x\leqslant b, c\leqslant y\leqslant d$。
- dom_st 取二元数组[a, b] 时，自变量范围是 $a\leqslant s\leqslant b, a\leqslant t\leqslant b$(假设参量是 s, t)；
  dom_st 取四元数组[a,b,c,d] 时，自变量范围是 $a\leqslant s\leqslant b, c\leqslant t\leqslant d$(假设参量是 s, t)。
- ngrid 是用来指定绘图格点数的。格点愈多，曲面表现愈细腻。默认时，ngrid=60。

● 输入宗量'circ' 指定图形在"圆域"上绘制。圆域为极坐标系。圆域中心在 $\left[\frac{a+b}{2},\frac{c+d}{2}\right]$处，半径 $r_m\sqrt{\left(\frac{b-a}{2}\right)^2+\left(\frac{d-c}{2}\right)^2}$。

● ezsurf 绘图时会自动标识图名、轴名。但用户也可用 title，xlabel 等指令重写所需的名称。

● 所有修饰指令，如 text，colormap，shading，light 等都可以用于 ezsurf 所画的图形。

● 其他在三维空间中绘制曲面的指令 ezmesh，ezcontour 等的使用方法，与 ezsurf 相似。

例【2.8-2】 使用球坐标参量画部分球壳（图 2.8-2）。

```
clf
x='cos(s)*cos(t)';                     % 两个独立参量描写 x
y='cos(s)*sin(t)';                     % 两个独立参量描写 y
z='sin(s)';                            % 两个独立参量描写 z
ezsurf(x,y,z,[0,pi/2,0,3*pi/2])        % 0≤s≤0.5π,0≤t≤1.5π
view(17,40)                            % 观察视角控制
shading interp
colormap(spring)                       % 绘制图形所用色图的设置
light('position',[0,0,-10],'style','local')    % 灯光控制
light('position',[-1,-0.5,2],'style','local')
material([0.5,0.5,0.5,10,0.3])                 % 曲面质感控制
```

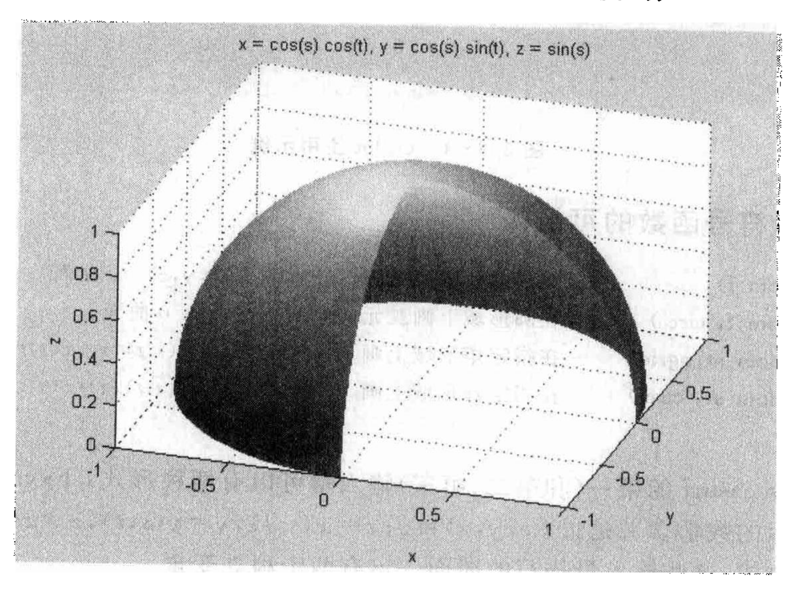


图 2.8-2 ezsurf 在参变量格式下绘制的图形

说明

● 与 ezsurf 参变量调用格式相似的指令还有 ezmesh，ezmeshc，ezsurfc 等。

● 在三维空间中，使用参变量画曲面时，每个"维度"变量都应表达为"两个参变量"的函数。

2.8.2 符号计算结果的数值化绘图

先把符号计算结果数值化，然后再利用 MATLAB 丰富的绘图指令实现可视化，也是比较常用的一种方法。以下通过算例进行演示。

例【2.8-3】 符号法求函数 $y=f(x)=1-\frac{2}{1+e^x}$ 的积分，用 plot 指令绘制函数及其积分函数的曲线；对反函数积分的两种算法进行可视化比较。本例演示：从符号计算结果产生双精度数值化数据；了解反函数性质；计算反函数积分的互补法。

(1) 函数 $y=f(x)=1-\frac{2}{1+e^x}$ 及其积分函数 $\int_0^x f(x)\mathrm{d}x$

```
clear
syms x y real                         % 假设在实数域中
fx = 1 - 2/(1 + exp(x));              % 创建函数
disp('f(x) = ')
pretty(fx)
disp('  ')
fxint = int(fx,x,0,x)                 % 计算积分函数
f(x) =


           2
  1 - --------------
       exp(x) + 1


fxint =
log((exp(x) + 1)^2/4) - x
```

(2) 从符号结果获得数值绘图数据，并绘图 2.8-3

```
xk = 0:0.1:2;                         % 指定自变量 x 的双精度数值采样点数组 xk
fxk = subs(fx,x,xk);                  % 获得对应的 f(x)双精度数组
fxintk = subs(fxint,x,xk);            % 对应 xk 数组求"f(x)关于 x 的积分数组"
plot(xk,fxk,'g:',xk,fxintk,'r','LineWidth',2.5)
                                      % 用双精度数组绘制曲线。注意:plot 只接受数值数据
title('函数及其积分函数')
xlabel('x')
legend('f(x)','\int^x 0 f(x) dx','Location','best')
```

(3) 求反函数 $x=g(y)$

```
gy = subs(finverse(fx),x,y)        % 求 f(x)的反函数 g(y)
gyint = int(gy,y,0,y)              % 求 g(y)的积分函数
gy =
log( - (y + 1)/(y - 1))
gyint =
piecewise([y < 1, log(1 - y^2) + y*log(y + 1) - y*log(1 - y)], [1 <= y, log(y^2 - 1) +
y*log( - (y + 1)/(y - 1)) - pi*i])
```

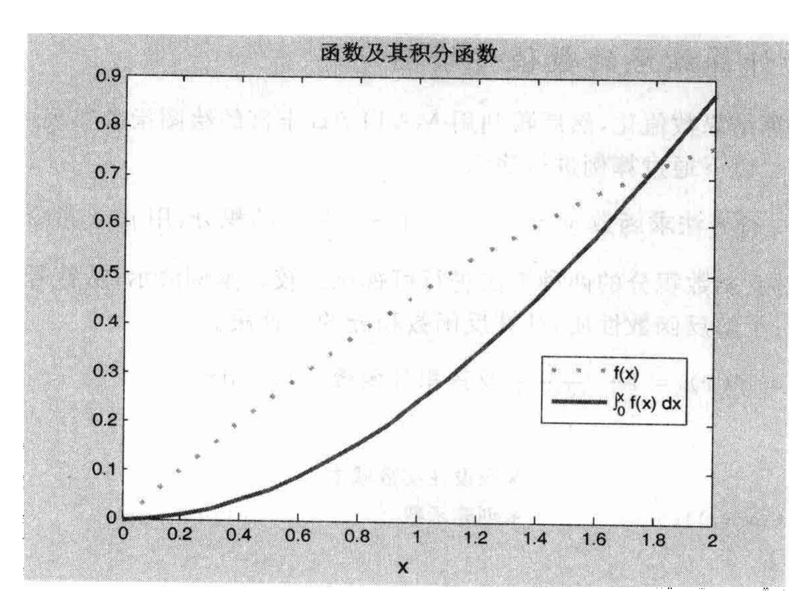


图 2.8-3 用双精度数据绘曲线

(4) 关系一：函数和反函数的互反性 $g(f(x))=x, f(g(y))=y$

```
gf = simplify(subs(gy,y,fx))
gf =
x
```

(5) 关系二：函数积分 $\int_a^x f(t)\mathrm{d}t$ 与反函数积分 $\int_{f(a)}^{f(x)} g(y)\mathrm{d}y$ 是关于"自变量与函数值所围矩形面积" $xf(x)$ 互补的，即 $\int_a^x f(t)\mathrm{d}t + \int_{f(a)}^{f(x)} g(y)\mathrm{d}y = xf(x)$，参见图 2.8-4。

```
yk = subs(fx,x,xk);                  % 对应 xk 的 yk 数组
gyintk = subs(gyint,y,yk);           % 直接法求"g(y)关于 y 的积分数组"
GYintk = xk. * fxk - fxintk;         % 借助互补关系计算反函数积分
plot(yk,gyintk,'r')
hold on
plot(yk,GYintk,' + k')
hold off
xlabel('y')
legend('直接法计算反函数积分','互补法求反函数积分','location','best')
```

说明

- 在 MATLAB 中，与 ezplot 相比，plot 指令对绘图对象的操控能力更强。因此，利用符号计算结果产生 plot 所需的双精度数据绘图是较常遇到的一种操作。在本书中，类似处理的还有例
- 本例介绍的"反函数求积的互补法"是解决"隐式反函数积分"的一种有效途径。比如，$f(x)=2-\frac{1}{1+\mathrm{e}^{-x}}-\frac{1}{1+\mathrm{e}^{-(x-1)}}$ 反函数的积分，若不用互补法求，恐怕会有点麻烦。

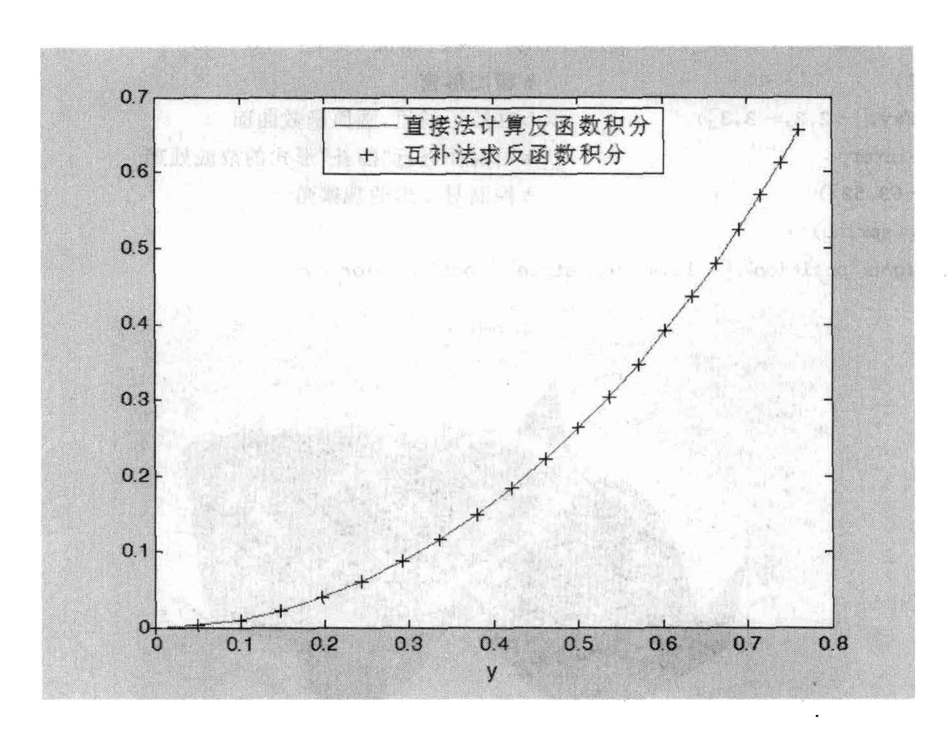


图 2.8-4 反函数积分两种算法结果比较

2.8.3 可视化与数据探索

例【2.8-4】 借助可视化手段，加深 Taylor 展开的邻域近似概念。图形研究函数 $f(x,y)=\sin(x^2+y)$ 在 $x=0,y=0$ 处的截断 8 阶小量的 Taylor 展开(例 2.3-7 讨论过)。本例演示：(A)通过对较大范围内的原函数图形和展开式图形观察，可以感受到两者之间完全不同的形态；(B)通过在 $x=0,y=0$ 小范围内原函数图形和展开式图形观察，可以感受到两者之间形态比较接近；(C)通过 $x=0,y=0$ 处的误差函数图形，可以观察到 Taylor 展开在“邻域”内的近似性能；(D)小字符串如何合并成大字符串；(E)ezsurf 的“圆域”绘图格式。

(1) 计算 Taylor 展开

```
TL1 = evalin(symengine,'mtaylor(sin(x^2 + y),[x,y],8)') % 字符串类型计算结果
TL1 =
(x^6 * y^2)/12 - x^6/6 + (x^4 * y^3)/12 - (x^4 * y)/2 - (x^2 * y^6)/720 + (x^2 * y^4)/24 - (x^2 * y
^2)/2 + x^2 - y^7/5040 + y^5/120 - y^3/6 + y
```

(2) 构造原函数和误差函数

```
Fxy = sym('sin(x^2 + y)')
Fxy_TL1 = Fxy - TL1
Fxy =
sin(x^2 + y)
Fxy_TL1 =
sin(x^2 + y) - y + (x^2 * y^2)/2 - (x^2 * y^4)/24 - (x^4 * y^3)/12 + (x^2 * y^6)/720 - (x^6 * y^
2)/12 + (x^4 * y)/2 - x^2 + x^6/6 + y^3/6 - y^5/120 + y^7/5040
```

(3) 在 $-4\leqslant x\leqslant 4$，$-6\leqslant y\leqslant 6$ 范围内的原函数曲面（见图 2.8-5）

```
figure(1)                                          % 清图形窗
ezsurf(Fxy,[-2,2,-3,3])                            % 在矩形域里，画原函数曲面
shading interp                                     % 对曲面进行"插补"形式的浓淡处理
view([-63,52])                                     % 控制对图形的观察角
colormap(spring)
light,light('position',[-10,4,50],'style','local','color','r')
```

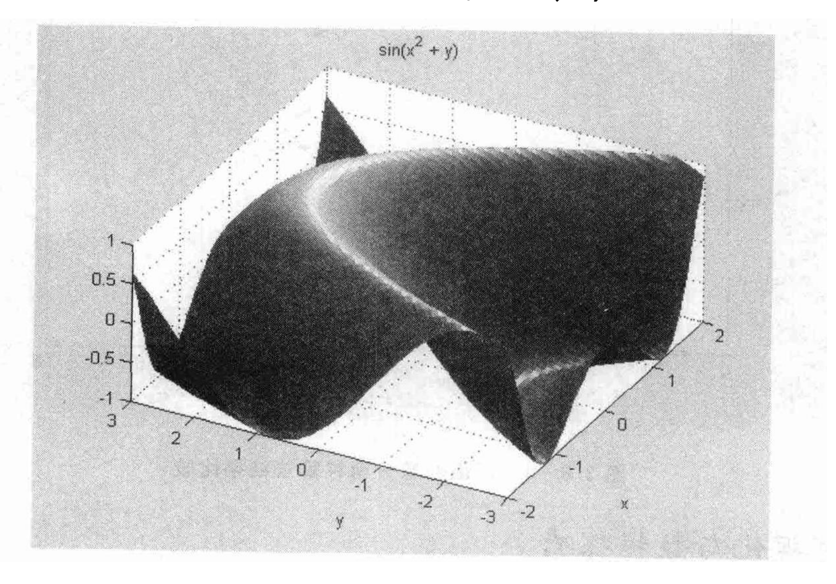


图 2.8-5 原函数在较大范围内的图形

(4) 在 $-4\leqslant x\leqslant 4$，$-6\leqslant y\leqslant 6$ 范围内的 Taylor 展开曲面（见图 2.8-6）

```
figure(2)
ezsurf(TL1,[-2,2,-3,3])
shading interp
view([-43,54])
colormap(spring)
light
light('position',[-10,2,2],'style','local','color',[0.8,0.3,0.3])
light('position',[-2,-10,2],'style','local','color',[0.4,0.5,0.7])
```

(5) 在 $-0.5\leqslant x\leqslant 0.5$，$-0.5\leqslant y\leqslant 0.5$ 范围内的原函数曲面（见图 2.8-7）

```
figure(3)
ezsurf(Fxy,[-0.5,0.5,-0.5,0.5],'circ')            % 圆形域绘图格式
axis([-1,1,-1,1,-2,2])
shading interp
colormap(spring)
view([-49,17])
light
light('position',[-30,0,-2],'style','local','color','r')
```

(6) 在 $-0.5\leqslant x\leqslant 0.5$，$-0.5\leqslant y\leqslant 0.5$ 范围内的误差曲面（见图 2.8-8）

```
figure(4)
ezsurf(Fxy TL1,[-0.5,0.5],'circ')
shading interp
colormap(spring)
view([-53,34])
light
light('position',[-10,15,0],'style','local','color',[0.8,0.3,0.3])
```

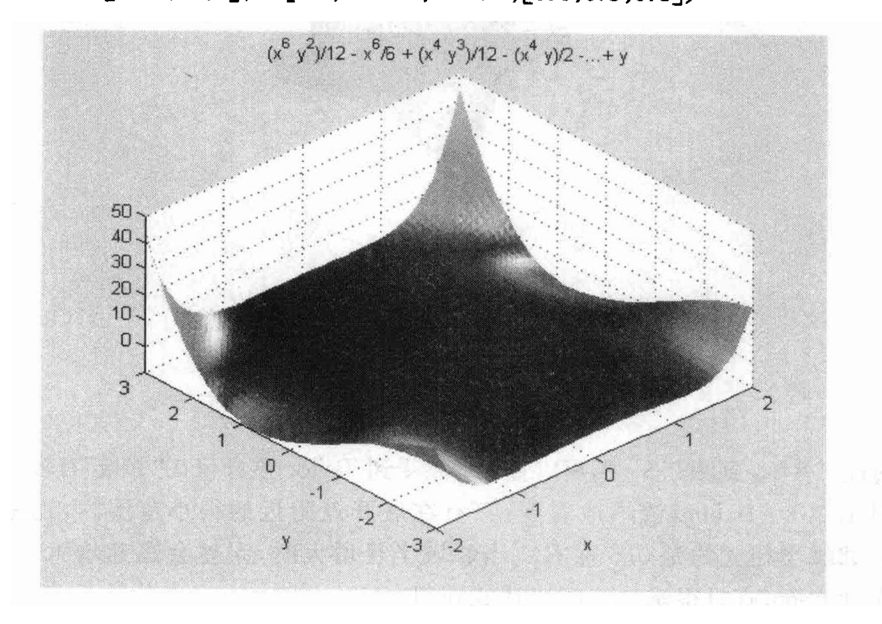


图 2.8-6 Taylor 展开在较大范围内的图形

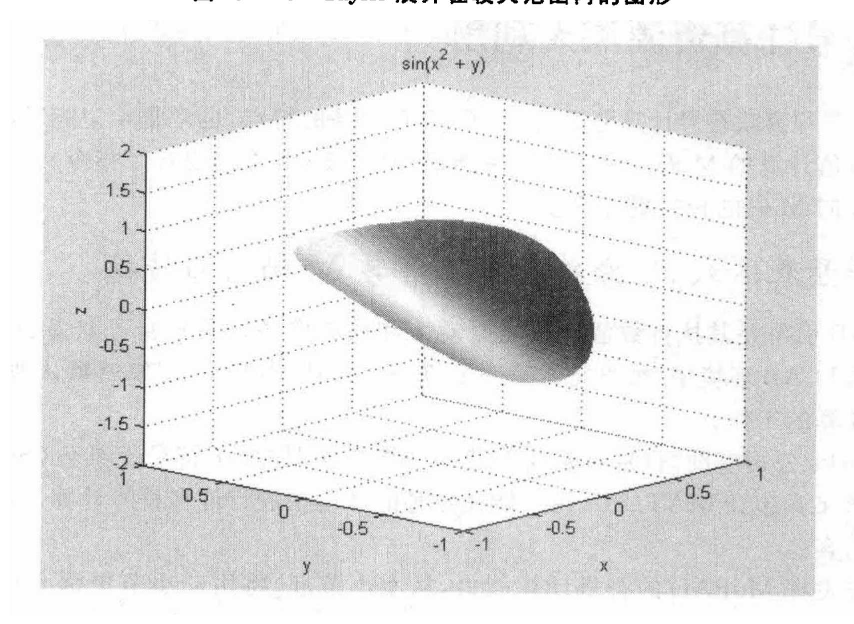


图 2.8-7 原函数在小范围内的图形

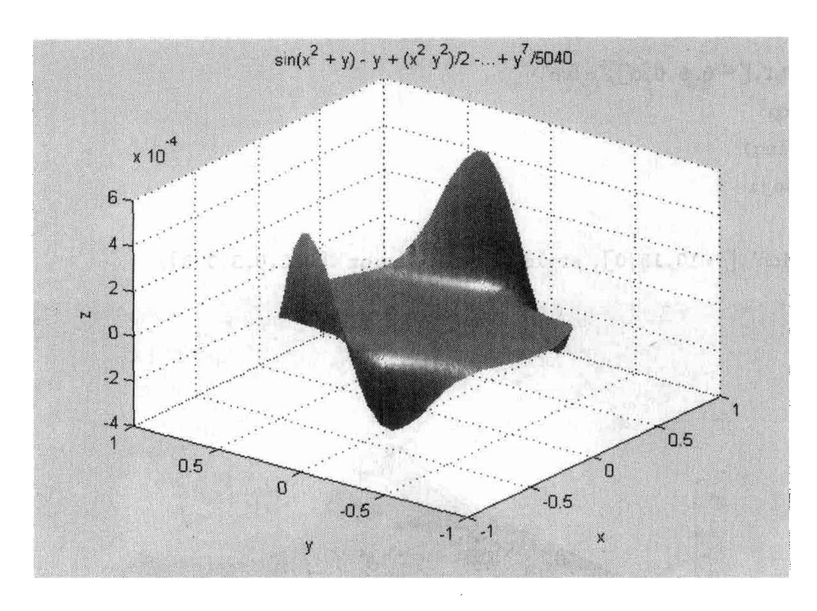


图 2.8-8 小范围内的误差曲面

说明

- 通过图 2.8-5 到图 2.8-8，可以直观地感受到 Taylor 展开仅在“邻域”有较好的近似性能。
- 通过图 2.8-8，可以清晰地看到：只有在展开点附近的很小范围内，近似是比较准确的。邻域半径大约是 0.5 左右。当邻域半径增大时，误差会迅速增大。值得指出：这种定性判断，有时很难从解析表达式获得。

2.9 符号计算资源深入利用

科技工作者很喜欢符号计算那简洁、明了、亲和力强的特点，也常常希望能通过符号计算途径写出适于数值计算的 M 码。为满足这种需求，MATLAB 专门设计了指令 matlabFunction，使符号表达式向 M 码的转换更加简便。

2.9.1 符号表达式、串操作及数值计算 M 码间的转换

MATLAB 引擎是其执行数值计算的原动力和核心资源。其本身不具备符号计算能力。在开启的 MATLAB 环境中，每当定义第一个符号对象时 MuPAD 引擎就默认地被启动，并由它承担符号计算的工作。

MathWorks 公司借助“符号对象及重载的计算方法”构成了符号工具包(Symbolic Math Toolbox)。该工具包让 MATLAB 用户能在熟悉的环境中进行许多符号计算。这就是此前八节所阐述的内容。

为进一步发挥 MuPAD 的符号计算能力，从本小节起，将用三小节更深入地介绍 MATLAB 与 MuPAD 两大引擎间的交互。由于这种交互离不开“字符串操作”，为此，本小节归纳性地罗列“符号计算、串操作、数值 M 码”间的相互联系：图 2.9-1 勾画了符号计算码与数值计算 M 码间的转换指令；表 2.9-1 罗列了本书所涉相关指令的节次和算例序号。

由本教材宗旨决定，图 2.9－1 和表 2.9－1 中所列的若干指令，将只是提及，而不做更深介绍。有兴趣的读者可以参见文献[1]。

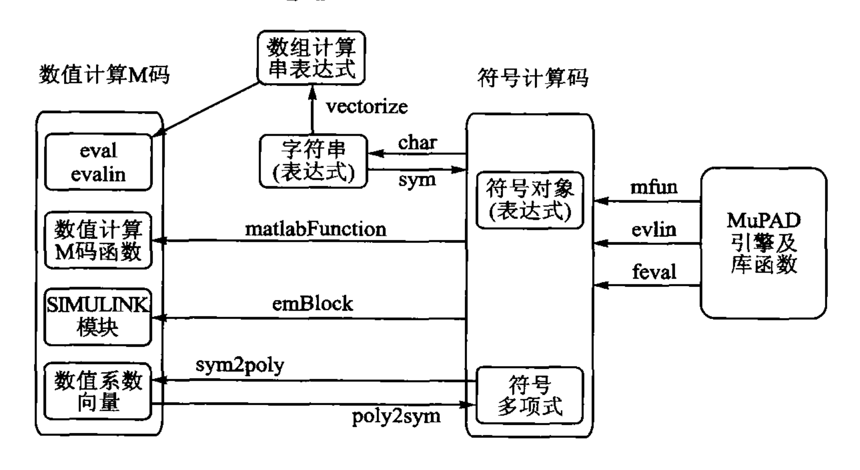


图 2.9－1 符号计算码与数值计算 M 码之间的转换指令

表 2.9－1 各指令的参考节次表

指令	参考节次	指令	参考节次
char	2.9—2，附录 A.1	poly2sym	
eval		sym	2.1，2.2；例 2.1－1，2.1－2，2.1－3，2.1－6，2.1－7
evalin	2.1－5；例 2.1－6，例 2.1－7	sym2poly	
matlabFunction	2.9.2；例 2.9－1，例 2.9－2	vectorize	2.9.2
mfun	2.9.3；例 2.9－3		

2.9.2 符号工具包资源表达式转换成 M 码函数

在科学研究和工程应用中，读者若希望在数值计算环境中利用符号计算的结果表达式，那末最简捷的途径之一是：借助 matlabFunction 指令把符号计算结果转换成 M 码匿名函数或 M 函数文件。

1. 转换指令 matlabFunction

Hmf = matlabFunction (f , param1, value1, ...)

把符号表达式转换成“函数句柄”、“匿名函数”、或 M 函数文件

说明

- 这是最通用的格式，“param－value 输入量对”最多可以有三对，但每对都可以省缺；输出量 Hmf 也可以省缺。它们省缺时的意义如表 5.9－2 所列。注意：第一输入量 f 是必不可少的。
- 第一输入量 f 必须是符号类表达式或代表符号表达式的符号变量。该表达式可能的来源是：

■ 由基本符号变量构成的“衍生”符号表达式；
■ 由 sym 指令作用于字符串表达式形成的符号表达式；
■ 被赋以符号表达式的符号变量；
■ 由 evalin 或 feval 运行 MuPAD 函数后获得的符号表达式。

● 第二输入量 param1 和第三输入量 value1 构成所谓的“param-value 输入对”。param 必须是字符串，它们可以是表 2.9-2 所列的三个关键词中的一个；而 value 的书写必须与所取关键词相应，如表 2.9-2 所列。

● Hmf 将是生成的 M 函数文件或匿名函数的“函数句柄”。假如 matlabFunction 调用时，不向 Hmf 赋值，那生成的函数句柄用缺省名 ans 。当然，ans 是临时的。一旦，其后再次运行别的“M 码表达式指令”，ans 就将被改写。

● matlabFunction 函数的工作机制是：先借助 char 把“符号表达式”变为“字符串”，再由 vectorize 把“字符串”变为“符合数组运算规则”的 M 码表达式。值得指出：若读者能理解这个工作机制，则对正确使用 matlabFunction 指令大有好处。

表 2.9-2 matlabFunction 的“param-value 输入对”的关键词及可能取值

<table>
<tr><th colspan="2">param 可取得关键词及含义</th><th colspan="2">value 的可能形式及举例</th></tr>
<tr><td rowspan="3">'file'</td><td rowspan="2">生成 M 函数文件</td><td>用带路径的文件名字符串，在指定目录上生成指定名的 M 文件。</td><td>'C:\Work\myfile'
在 C 盘 Work 目录上生成 myfile. m</td></tr>
<tr><td>不带路径的文件名字符串，在当前目录上生成指定名 M 文件。</td><td>'myfile'
在当前目录上生成 myfile. m</td></tr>
<tr><td colspan="3">没有关键词'file'构成的"param-value 输入对"，则生成匿名函数</td></tr>
<tr><td rowspan="3">'vars'</td><td rowspan="2">指定生成文件或匿名函数中输入量的次序</td><td>符号变量数组
或符号变量的胞元数组</td><td>[z, y, x]
注意：z, y, x 必须是已定义的符号变量</td></tr>
<tr><td>字符串胞元数组</td><td>{'y', 'x', 'z'}
可以是事先未定义变量名</td></tr>
<tr><td colspan="3">没有关键词'vars'构成的"param－value 输入对"，则按字母表次序排列变量名</td></tr>
<tr><td rowspan="3">'outputs'</td><td>指定生成文件或匿名函数中输出量的次序</td><td>字符串胞元数组</td><td>{'v1', 'v2', 'v3'}</td></tr>
<tr><td rowspan="2">没有关键词'outputs'构成的"param-value 输入对"</td><td>若第一输入量是被赋以符号表达式的符号变量，则生成的 M 码文件以此符号变量名作为输出量名。</td><td>如 f 是 matlabFunction 的第一输入量，则生成的 M 文件的输出量名就是 f</td></tr>
<tr><td>若第一输入量为符号表达式，则生成的 M 函数文件的输出量名为 out1</td><td></td></tr>
</table>

2. 把符号包资源转换成 M 码函数的示例

本小节将通过算例具体描述，使用 matlabFunction 创建 M 函数文件和匿名函数的基本步骤和注意事项。

例【2.9-1】 试写出能对于任给(具体数值的)边界条件，计算微分方程 $xy''-3y'=x^2$ 数值解的 M 码。本例目的：演示 matlabFunction 的"完善"调用格式；由 matlabFunction 所生成的 M 文件及其必须进行的注释完善化。

（1）求微分方程两点边值问题的解

已知两点边值分别为 $y(a)=y_L$ 和 $y(b)=y_R$，用符号法解微分方程如下：

```
s = dsolve('x * D2y - 3 * Dy = x^2','y(a) = yL,y(b) = yR','x')
s =
(a^4 * b^3 + 3 * yR * a^4 - a^3 * b^4 - 3 * yL * b^4)/(3 * a^4 - 3 * b^4) - x^3/3 + (x^4 * (a^3 - b^3
 + 3 * yL - 3 * yR))/(3 * a^4 - 3 * b^4)
```

（2）创建"在任意给定的 a,b,y_L,y_R 下计算微分方程数值解"的 M 函数文件

```
Hs = matlabFunction(s,'file','exm020901 auto','vars',{'x','a','b','yL','yR'},'outputs',{'y'})
                                                                                          % <2>
Hs =
    @exm020901_auto
```

（3）验证文件的正确性

为与例 2.4-3 比较，设边界条件为 $y(1)=0,y(5)=0$，画出方程在 $-1\leqslant x\leqslant 6$ 区间的解曲线(见图 2.9-2)。具体如下。

```
a = 1;b = 5;yL = 0;yR = 0;
xn = -1:6;yn = Hs(xn,a,b,yL,yR)              % 专供数值比较而算                    <4>
x = -1:0.2:6;y = Hs(x,a,b,yL,yR);            % 为把解曲线画得较光滑而算            <5>
plot(x,y,'b-'),hold on
plot([1,5],[0,0],'.r','MarkerSize',20),hold off
title(['xy{\prime\prime} - 3y{\prime} = x^2,',' ','y(1) = 0, y(5) = 0'])
text(1,1,'y(1) = 0'),text(4,1,'y(5) = 0')
xlabel('x'),ylabel('y')
yn =
  Columns 1 through 7
    0.6667    0.2671    0.0000   -1.3397   -3.3675   -4.1090    0.0000
  Column 8
   14.1132
```

（4）完善所建文件的注释

- 由 matlabFunction 所创建的 exm020901_auto.m 函数 M 文件如下。该文件的注释过于简单，甚至连关于函数数学性质、输入输出量的起码说明都没有。

```
function y = exm020901_auto(x,a,b,yL,yR)
%EXM020901 AUTO
%    Y = EXM020901 AUTO(X,A,B,YL,YR)
```

```
%     This function was generated by the Symbolic Math Toolbox version 5.4.
%     22 - Apr - 2010 15:07:03

t54 = a.^2;
t55 = b.^2;
t56 = t54.^2;
t57 = t55.^2;
t58 = x.^2;
t59 = t56. * 3.0;
t60 = t57. * 3.0;
t61 = t59 - t60;
t62 = 1.0./t61;
y = - t62. * (t57. * yL. * 3.0 - t56. * yR. * 3.0 + a. * t54. * t57 - b. * t55. * t56) -
t58. * x. * (1.0./3.0) + t58.^2. * t62. * (yL. * 3.0 - yR. * 3.0 + a. * t54 - b. * t55);
```

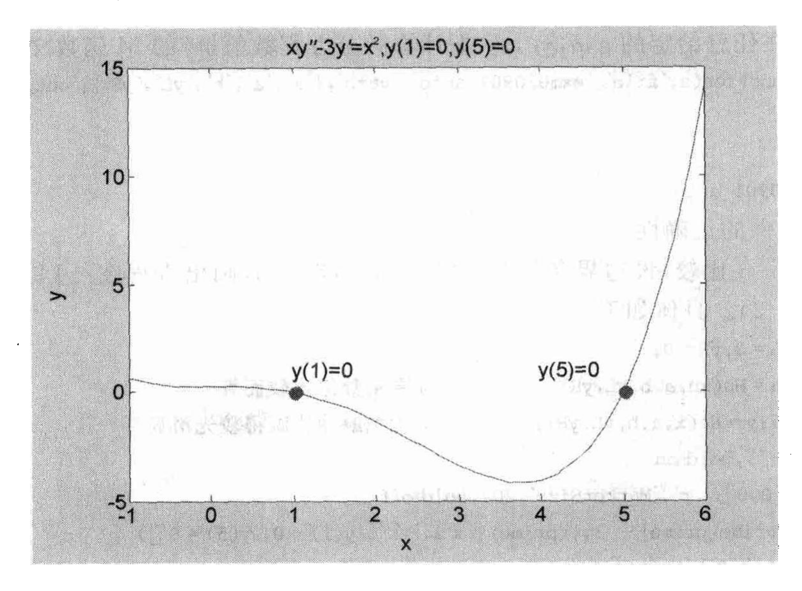


图 2.9-2 两点边值问题的解曲线

● 为 M 文件今后使用的方便，应该对自动生成的 M 文件进行注释，及时地加以完善。下面列出：对 exm020901_auto.m 采用“另存为”操作；再经注释完善后的”函数文件 exm020901_ZZY.m。注释区中，“楷体”文字是本书作者手工加入的。

```
function y = exm020901_ZZY(x,a,b,yL,yR)
% EXM020901_ZZY
%  Y = EXM020901_ZZY(X,A,B,YL,YR)
%                                   两点边值问题的数值解
%  该两点边值问题是      x * (D2y)^2 - 3 * (Dy) = x^2, y(a) = yL, y(b) = yR
%  x           自变量数值数组
%  a, b        分别是左、右边值的 x 坐标
%  yL, yR      分别是左、右边值的 y 坐标
```

```
%    This function was generated by the Symbolic Math Toolbox version 5.4.
%    25 - Mar - 2010 17:13:43

t7 = a.^2;
t8 = b.^2;
t9 = t7.^2;
t10 = t8.^2;
t11 = x.^2;
t12 = 3. * t10;
t13 = t12 - 3. * t9;
t14 = 1./t13;
y = t14. * (3. * t10. * yL - 3. * t9. * yR + a. * t10. * t7 - b. * t8. * t9) - (x. * t11)./3 -
t14. * (3. * yL - 3. * yR + a. * t7 - b. * t8). * t11.^2;
```

（5）直接利用 M 函数文件名计算两点边值问题的数值解

```
ym = exm020901_ZZY(xn,a,b,yL,yR)                                         % <12>
ym =
  Columns 1 through 7
    0.6667    0.2671    0.0000    -1.3397    -3.3675    -4.1090    0.0000
  Column 8
   14.1132
```

说明

- 本例采用了 matlabFunction 的完整调用格式。
  - 文件名被指定为 exm020901_auto。因为没有指定路径，所以文件建立在当前目录上。
  - 所创建的文件中的自变量名称和次序都是由 matlabFunction 中'vars'参数的值{'x','a','b','yL','yR'}指定的。注意：由于在本例中，符号对象 s 中的各变量、参数都未定义过，所以必须采用这种格式。
  - 文件的输出变量 y，则是由'outputs'的值{'y'}指定的。
  - 本例中，matlabFunction 运行后产生的函数句柄赋给了 Hs。因此，在第〈4〉〈5〉条指令中可以调用 Hs 计算函数值 y。
- “函数句柄”（见第〈4〉〈5〉条指令）和“M 函数文件名”（见第〈12〉条指令）都可以用于计算函数 y 的数值解。但请注意：
  - 函数句柄并不永久存在。一旦 MATLAB 内存被 clear 清除，或 MATLAB 被关闭后再次启动，原先建立的函数句柄不再存在。
  - M 函数文件是永久驻留在硬盘上的，除非故意将它删除。

例【2.9-2】 编写一个通用的求解微分方程两点边值问题数值解的 M 函数文件。本例目的：演示如何综合运用符号计算和数值计算，编写带某种通用性的微分方程两点边值问题解算程序；演示 matlabFunction“产生匿名函数”的调用格式。

（1）编写文件

```
function y = exm020902_ZZY(de,x,a,b,yL,yR,flag)
```

```
% EXM020902 ZZY
% y = exm020902 ZZY(de,x,a,b,yL,yR) 计算两点边值问题的数值解
% de          必须是以 x 为自变量、y 为从变量的微分方程字符串
% x           自变量数值数组
% a, b   分别是左、右边值的 x 坐标
% yL, yR   分别是左、右边值的 y 坐标
% flag   flag 取 0,不画解曲线;flag 取 1,则画解曲线。
% 25 - Mar - 2010 17:25:10 by ZZY

if nargin~ = 7
    error('输入量数目应有 7 个 !')
end
s = dsolve(de,'y(a) = yL,y(b) = yR','x');
Hs = matlabFunction(s,'vars',{'x','a','b','yL','yR'});                  % <14>
y = Hs(x,a,b,yL,yR);
if flag = = 1
    plot(x,y,' - b',[a,b],[yL,yR],' * r')
    title(de)
    xlabel('x'),ylabel('y')
    shg
end
```

(2) 验证性计算

采用例 2.9 - 1 的微分方程和边值条件,调用本例提供的 M 文件 exm020902_ZZY. m,进行如下验证性计算(参见图 2.9 - 3)。

```
x = - 1:6;
a = 1;b = 5;yL = 0;yR = 0;
de1 = 'x * D2y - 3 * Dy = x^2';
y = exm020902 ZZY(de1,x,a,b,yL,yR,0)
y =
    0.6667    0.2671    0.0000    - 1.3397    - 3.3675    - 4.1090         0    14.1132
```

(3) 对其他微分方程的应用

用 exm020902_ZZY. m 文件试求 $y(3)=-2, y(6)=10$ 条件下微分方程 $x''-3y'=3x^2+x$ 的解。

```
de2 = 'x * D2y - 3 * Dy = 3 * x^2 + x';
x = - 1:0.2:6;
y = exm020902 ZZY(de2,x,3,6, - 2,10,1);
```

说明

- 对解算微分方程边值问题来说,符号解法显得比数值解法简洁明了;但是,数值解法的适应性要比符号法强得多。相当多的微分方程,如若用符号法求解,或因找不到“封闭解”而失败,或因计算时间过长而不得不放弃。
- 本例演示的是:当科研需要对某类微分方程求解时,如何编写带某种通用性的程序。

● exm020902_ZZY.m 文件第〈14〉条指令，由于没有 'file'选项及其相应的值，所以 matlabFunction 运行后就生成匿名函数，并把该匿名函数的句柄存放在变量 Hs 中。

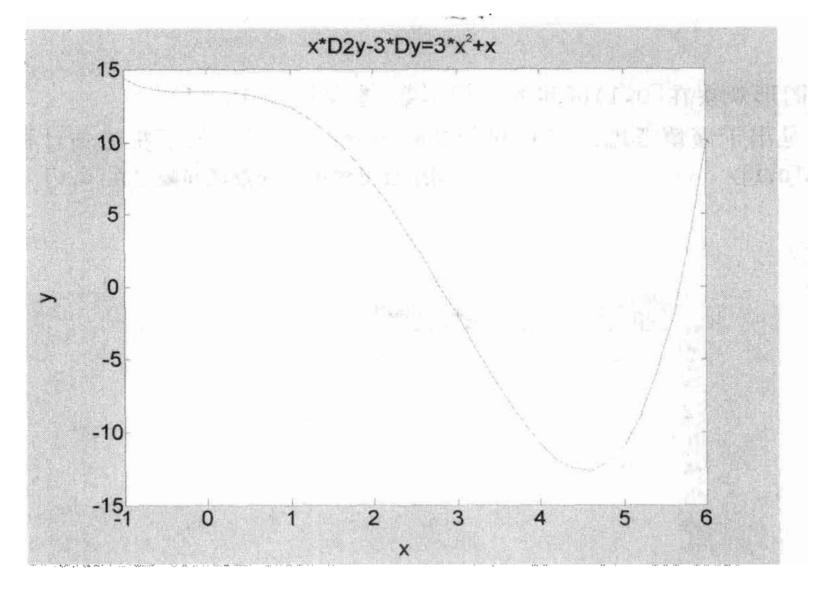


图 2.9-3 用 exm020902_ZZY.m 求解另一微分方程所得解

2.9.3 借助 mfun 调用 MuPAD 特殊函数

出于“科学研究和工程应用需要”和“MATLAB 数值计算基础库中短缺”的原因，在 MATLAB 的符号计算包(Symbolic Math Toolbox)就专门设计了 mfun 接口文件，以实现“在 MATLAB 中直接利用 MuPAD 特殊函数”完成数值计算。

关于 MuPAD 特殊函数的列表可用 mfunlist 获得，而更详细的关于各特殊函数指令的调用格式可以用 mfunlist 在 MATLAB 帮助浏览器中搜索得到(参见第 5.1～5.5 节)。下面介绍借助 mfun 调用特殊函数指令的格式。

fx=mfun(Fname, par1, …, par4)　　　　**调用特殊函数进行数值计算**

说明

● Fname 必须是由 mfunlist 查得的特殊函数名的字符串。

● par1，…，par4 必须是数值；每个参数含义、参数的具体数目和次序必须与 mfunlist 列表一致。

● 计算所得结果 fx 是 16 位数字精度的数值。

例【2.9-3】 已知 $f(x)=\int_0^x \frac{1}{\ln t}dt$，试画出 $f(x)$在 $0\leqslant x\leqslant 0.9$ 的曲线。本例演示：用特殊函数表示的计算结果的判读；mfun 指令的调用格式及注意事项；计算前的准备性评估和计算后的验证性判断；MATLAB 函数 expint 与 MuPAD 函数 Li(x)的对照。

(1) 检查被积函数在 $t=0^+$ 处的极限

因为被积函数中，$\ln t$ 在 $t=0$ 处无定义，出于谨慎，先计算被积函数在积分下限处的极限，以对积分可行性判断有所帮助。

```
syms t
```

```
gt=1/log(t);
gt_0=limit(gt, t,0,'right')        %考虑右极限是由题目决定的
gt_0 =
0
```

(2) 利用图形观察在[0,1)区间的被积函数(参见图 2.10-1)

该步骤也是出于谨慎考虑。它有利于判断积分的可行性;便于粗略估计积分结果。

```
ezplot(gt,[0,1])                   %据题目要求把自变量区间限定在[0,1]
grid on
legend('gt')
```

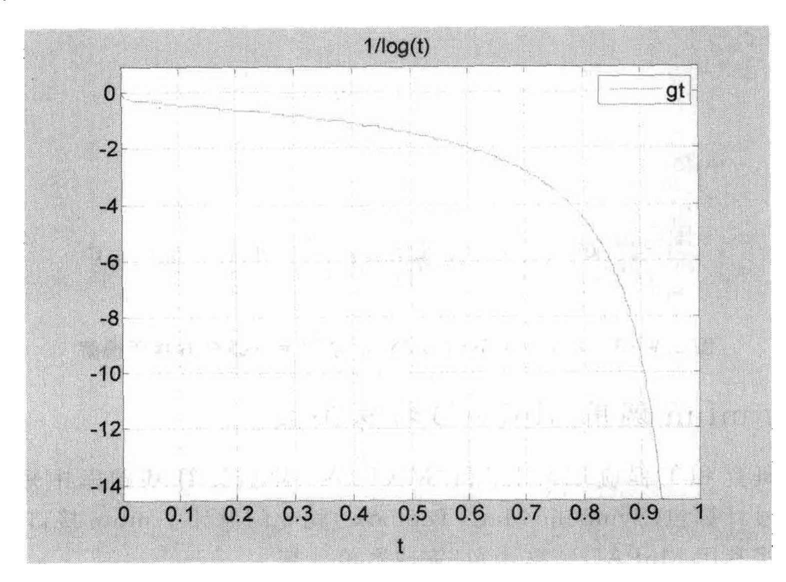


图 2.10-1 在关心区间内的被积函数

(3) 求被积函数的原函数

```
fx=int(gt,t,0,x)                                              %  <7>
Warning: Explicit integral could not be found.
fx =
piecewise([x < 1, Li(x)], [Otherwise, int(1/log(t), t = 0..x)])
```

(4) 利用 mfun 指令计算 x 不同取值时的定积分

```
x=0:0.05:0.9;          %X定义为数值数组,才能保证 log(x)是数值数组     <8>
fxMfun=mfun('Li',x)                                            %  <9>
                       %借助 mfun 调用 MuPAD 的 Ei 函数计算定积分
fxMfun =
  Columns 1 through 8
      NaN   -0.0131   -0.0324   -0.0564   -0.0851   -0.1187   -0.1574   -0.2019
  Columns 9 through 16
  -0.2529   -0.3114   -0.3787   -0.4564   -0.5469   -0.6534   -0.7809   -0.9369
  Columns 17 through 19
  -1.1340   -1.3959   -1.7758
```

(5) 在被积函数图上画积分曲线(参见图 2.10-2)

```
hold on
plot(x,fxMfun,'--r','LineWidth',3)
legend('gt','fxMfun','Location','Best')
title(' ')                    % 为了删除 ezplot 自动产生的图名
hold off
```

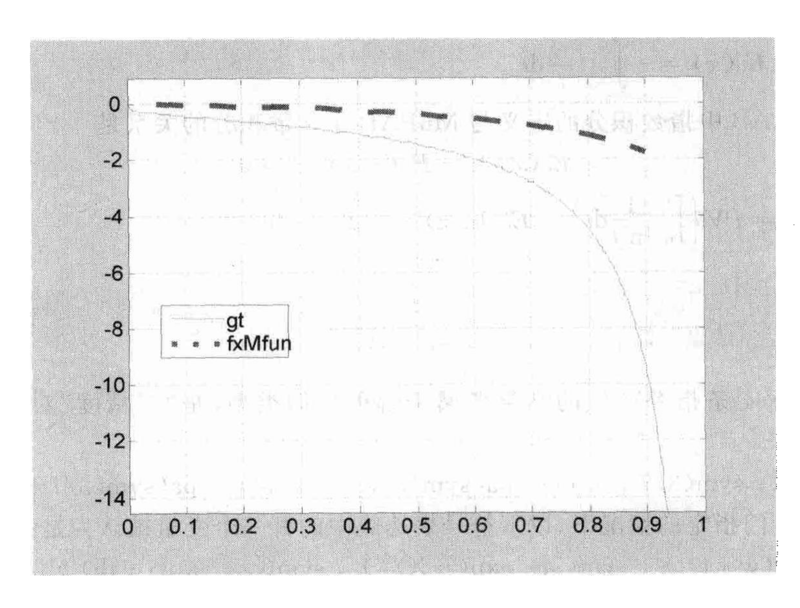


图 2.10-2 被积函数曲线 gt 和积分曲线 fx

(6) 调用 toolbox\matlab\specfun 文件夹上的 expint 指令计算本例积分

```
fx_matlab = -expint(-log(x))                                        % <15>
                        % 利用 MATLAB 自己的数值计算函数计算指数积分
fx_matlab =
  Columns 1 through 8
       NaN   -0.0131   -0.0324   -0.0564   -0.0851   -0.1187   -0.1574   -0.2019
  Columns 9 through 16
   -0.2529   -0.3114   -0.3787   -0.4564   -0.5469   -0.6534   -0.7809   -0.9369
  Columns 17 through 19
   -1.1340   -1.3959   -1.7758
```

说明

- 第〈7〉行指令给出的计算结果 piecewise([x < 1, Li(x)], [Otherwise, int(1/log(t), t = 0..x)]) 是 MuPAD 表达格式。它表示:积分结果是分段函数;在 $x<1$ 区间,积分可用"对数积分函数(Logarithmic integral)"Li(x) 表达。在 mfunlist 函数列表中,$Li(x) = PV\left\{\int_0^x \frac{1}{\ln t}\mathrm{d}t\right\} = Ei(\ln x)$ 。在此 PV 表示复数中的"主值"。
- 本例的第〈8〉〈9〉行指令是用来计算若干定积分数值的。第〈9〉行指令的编写依据是:
  - 因为 mfun 的第一个输入量要求"函数名字符串",所以根据前面的输出结果中的[x < 1, Li(x)],为计算数值结果,就写出第〈9〉行那样的指令。

- 又因为 mfun 要求除第一个输入量外的其他输入参数是“(双精度)数值”，所以在第〈8〉行定义了一个(双精度)数值数组 x。

- 在 MATLAB 的 toolbox\matlab\specfun 文件夹上也有一些计算特殊函数值的文件。其中包括计算“指数积分”的 expint 指令。写出第〈15〉条指令的根据是：
  - 在 MATLAB 中，expint 指令对指数积分的定义是 $E_1(x)=\int_x^{\infty}\frac{e^{-t}}{t}dt$。而 MuPAD 中定义 $Ei(x)=-\int_{-\infty}^{x}\frac{e^{t}}{t}dt$。
  - MATLAB 中指数积分的定义与 MuPAD 中指数积分的关系是

$$Ei(x)=-E_1(-x),x>0$$

  - $Li(x)=PV\left\{\int_0^x\frac{1}{\ln t}dt\right\}=Ei(\ln x)=-E_1(-\ln x)$

习题 2

1. 说出以下 4 条指令产生的结果各属于哪种数据类型，是“双精度”对象，还是“符号”对象？
   3/7+0.1; sym(3/7+0.1); vpa(sym(3/7+0.1),4); vpa(sym(3/7+0.1))
2. 在不加专门指定的情况下，以下符号表达式中的哪一个变量被认为是独立自由变量。
   sym('sin(w * t)') , sym('a * exp(-X)') , sym('z * exp(j * th)')
3. 求以下两个方程的解。(提示：关于符号变量的假设要注意。)
   (1) 试写出求三阶方程 $x^3-44.5=0$ 正实根的程序。注意：只要正实根。
   (2) 试求二阶方程 $x^2-ax+a^2=0$ 在 $a>0$ 时的根。
4. 观察一个数(在此用@记述)在以下四条不同指令作用下的异同：
   a = @ , b = sym(@), c = sym(@ ,'d'), d = sym('@ ')
   在此，@ 分别代表具体数值 7/3 , pi/3 , pi * 3^(1/3) ；而异同通过 vpa(abs(a-d)) , vpa(abs(b-d)) , vpa(abs(c-d))等来观察。
5. 求符号矩阵 $\boldsymbol{A}=\begin{bmatrix}a_{11} & a_{12} & a_{13}\\ a_{21} & a_{22} & a_{23}\\ a_{31} & a_{32} & a_{33}\end{bmatrix}$ 的行列式值和逆，所得结果应采用“子表达式置换”简洁化。
6. 对函数 $f(k)=\begin{cases}a^k & k\geqslant 0\\ 0 & k<0\end{cases}$，当 a 为正实数时，求 $\sum_{k=0}^{\infty}f(k)z^{-k}$。(提示：symsum。实际上，这就是根据定义求 Z 变换问题。)
7. 对于 $x>0$，求 $\sum_{k=0}^{\infty}\frac{2}{2k+1}\left(\frac{x-1}{x+1}\right)^{2k+1}$。(提示：理论结果为 $\ln x$；注意限定性假设。)
8. (1) 通过符号计算求 $y(t)=|\sin t|$ 的导数 $\frac{dy}{dt}$。(2) 然后根据此结果，求 $\left.\frac{dy}{dt}\right|_{t=0}$ 和 $\left.\frac{dy}{dt}\right|_{t=\frac{\pi}{2}}$。

9. 求出 $\int_{-5\pi}^{1.7\pi} e^{-|x|} |\sin x| dx$ 的具有64位有效数字的积分值。(提示:int, vpa, ezplot。)

10. 计算二重积分 $\int_1^2 \int_1^{x^2} (x^2 + y^2) dy dx$ 。

11. 在[0,2π]区间,画出 $y(x) = \int_0^x \frac{\sin t}{t} dt$ 曲线,并计算 $y(4.5)$。(提示:int, subs。)

12. 在 $n>0$ 的限制下,求 $y(n) = \int_0^{\frac{\pi}{2}} \sin^n x dx$ 的一般积分表达式,并计算 $y\left(\frac{1}{3}\right)$的32位有效数字表达。(提示:注意限定条件;注意题目要求32位有效。)

13. 有序列 $x(k) = a^k, h(k) = b^k$,(在此 $k \geqslant 0, a \neq b$),求这两个序列的卷积 $y(k) = \sum_{n=0}^{k} h(n)x(k-n)$ 。(提示:symsum, subs。)

14. 设系统的冲激响应为 $h(t) = e^{-3t}$,求该系统在输入 $u(t) = \cos t, t \geqslant 0$ 作用下的输出。(提示:直接卷积法,变换法均可。)

15. 求 $f(t) = Ae^{-a|t|}, \alpha > 0$ 的 Fourier 变换。(提示:注意限定。)

16. 求 $f(t) = \begin{cases} A\left(1 - \frac{|t|}{\tau}\right) & |t| \leqslant \tau \\ 0 & |t| > \tau \end{cases}$ 的 Fourier 变换,并画出 $A=2, \tau=2$ 时的幅频谱。(提示:注意限定;simple。)

17. 求 $F(s) = \frac{s+3}{s^3 + 3s^2 + 6s + 4}$ 的 Laplace 反变换。

18. 利用符号运算证明 Laplace 变换的时域求导性质:$L\left[\frac{df(t)}{dt}\right] = s \cdot L[f(t)] - f(0)$ 。(提示:用 sym('f(t)')定义函数 $f(t)$。)

19. 求 $f(k) = ke^{-\lambda kT}$ 的 Z 变换表达式。

20. 求方程 $x^2 + y^2 = 1, xy = 2$ 的解。(提示:正确使用 solve。)

21. 求图 p2-1 所示信号流图的系统传递函数,并对照胡寿松主编《自动控制原理》中的例 2-21 结果,进行局部性验证。(提示:在局部性验证时,把不存在支路的符号增益设置为0。)

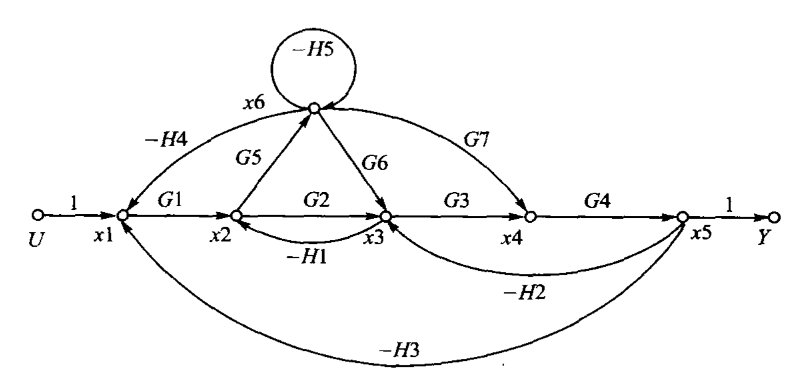


图 p2-1

22. 采用代数状态方程法求图 p2－2 所示结构框图的传递函数$\frac{Y}{U}$和$\frac{Y}{W}$。(提示:列出正确的状态方程$\begin{cases}x=Ax+bU+fW\\Y=ck+dU+gW\end{cases}$,进而写出相关输入输出对之间的传递函数表达式$\frac{Y}{U}=c(I-A)^{-1}b+d$和$\frac{Y}{N}=c(I-A)^{-1}f+g$)

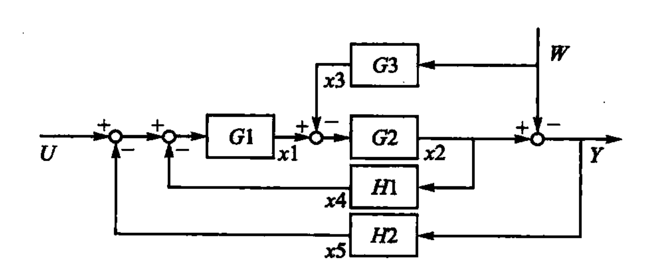


图 p2－2

23. 求微分方程 $yy'/5+x/4=0$ 的通解,并绘制任意常数为 1 时,如图 p2－3 所示的解曲线图形。(提示:通解中任意常数的替代;构造能完整反映所有解的统一表达式,然后绘图。)

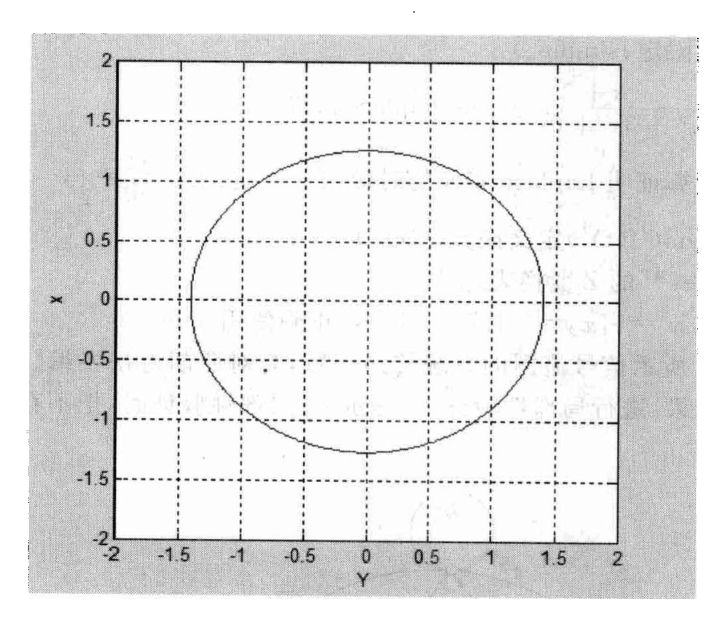


图 p2－3 微分方程的解曲线

24. 求一阶微分方程 $x'=at^2+bt, x(0)=2$ 的解。

25. 求边值问题$\frac{df}{dx}=3f+4g, \frac{dg}{dx}=-4f+3g, f(0)=0, g(0)=1$ 的解。

第3章 数值数组及向量化运算

数值数组(Numeric Array)和数组运算(Array Operations)始终是 MATLAB 的核心内容。本书从第 3 章起，全部注意力将集中于数值数组及其运算。

本章系统阐述：数组浮点算法的特点；一、二维数值数组的创建和寻访；数组运算和向量化编程；实现数组运算的基本函数；常用标准数组生成函数和数组构作技法；非数 NaN、“空”数组概念和应用；关系和逻辑操作。

3.1 数值计算的特点和地位

符号计算可以对包含变量字符、参数字符和数字的表达式进行推理、运算，并给出符号结果。这些结果与高校教科书中的解析表达式比较接近。这是符号计算的长处，也是它们至今仍在科学计算中保留一席之地的原因。然而，符号计算的明显短处是：有很多问题无法解，有很多问题求解时间过长。因此，在实际的科学计算、工程分析和设计中，符号计算的适用范围远远不如数值计算。

现代的主流计算机都只能表示、保存、运算和输出有限精度的数字。数值计算正适应了现代主流计算机的这种特点，它计算速度快，容量大，能处理各种复杂的函数关系。但由于数值计算以有限精度数字为基本操作元素，所以它只能用有限长度的数据，以有限的精度表现有限时间和范围内的函数关系。

下面用两个算例来说明，符号计算和数值计算的不同特点。

例【3.1-1】 已知 $f(t)=t^2\cos t$，求 $s(x)=\int_0^x f(t)\mathrm{d}t$。本例演示：(A) 本题通过人工方法可以得到“封闭解析表达式”，符号计算特别适宜；(B) 两种算法机理上的差异；(C) 数值计算的离散本质；(D) 图形有助于理解数值计算结果的函数意义。

(1) 符号计算解法

```
syms t x                % 定义符号变量——被保存的是符号
ft = t^2 * cos(t)       % 采用基本符号变量，构造出符号被积函数，并被保存在 ft
sx = int(ft,t,0,x)      % 在基本符号变量、符号表达式的基础上进行一系列的推理和
                        % 演绎，获得的一个用符号表达的原函数

ft =
t^2 * cos(t)
sx =
```

```
x^2 * sin(x) - 2 * sin(x) + 2 * x * cos(x)
```

(2) 数值计算解法

```
dt = 0.05;                 % 为在连续区间上取离散的计算点,先设定采样间隔
t = 0:dt:5;                % 根据题意,时间区间的左端取为 0。右端点也必须确定,如取 5
                           % 数值计算只能在有限的区间上,取有限个采样点
Ft = t.^2. * cos(t);       % 按函数关系,在有限个自变量采样点上,进行数值计算 f(t),
                           % 并把计算得到的数值以 Ft 变量名加以保存
                           % f(t)数据的总点数与 t 点数相同
Sx = dt * cumtrapz(Ft);    % 计算从 0 开始到每个采样点为止的区间内,Ft 曲线下的面积
                           % 这个面积是由一个个宽度为 dt 的小梯形面积累加而得的
t(end - 4:end)             % 在此 end 指示最后一个元素位置
Sx(end - 4:end)            % Sx 的最后 5 个元素
plot(t,Sx,'.k','MarkerSize',12)
                           % 利用如图 3.1-1 所示的图形,有助于了解所求原函数的性状
xlabel('x'),ylabel('Sx'),grid on
ans =
     4.8000    4.8500    4.9000    4.9500    5.0000
ans =
   -20.1144  -19.9833  -19.7907  -19.5345  -19.2131
```

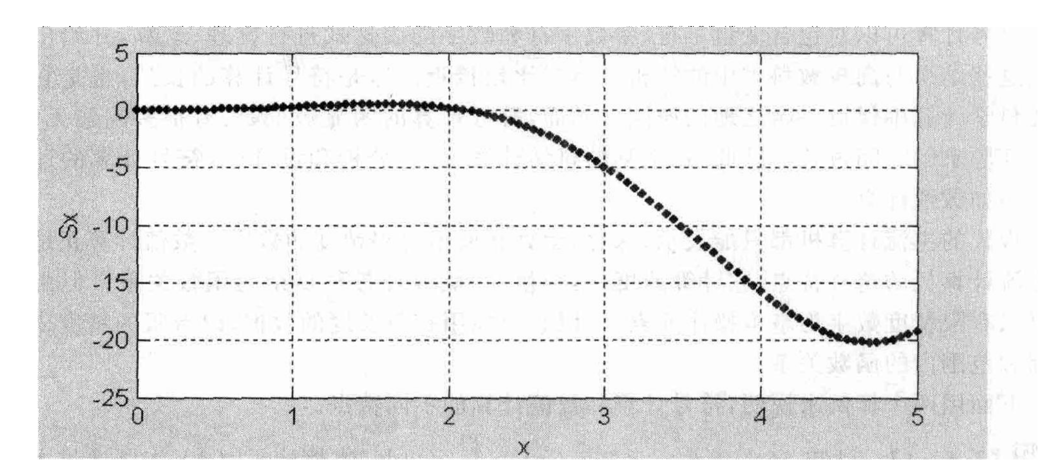


图 3.1-1 在区间[0, 5]采样点上算得的定积分值

说明

- 进行数值计算,必须首先确定一组自变量采样点。这意味着:一,确定一个自变量的取值区间;二,取定采样间隔(可以等间隔,也可以不等间隔)。
- 执行数值计算的表达式都是在已知的数值点上进行的。
- 数值计算结果也是离散的。
- 一般说来,直接观察数据,难以抽象出这组数据的内涵;而离散数据的图形曲线可以形象地体现数据间的函数关系。但要注意:图形展示的函数性状仅在自变量的取值区间有意义,任何对区间外的延伸和猜测都需特别谨慎。

例【3.1-2】 已知 $f(t)=e^{-\sin(t)}$，求 $s(x)=\int_0^4 f(t)\mathrm{d}t$。本例演示：(A) 本题被积函数的原函数没有“封闭解析表达式”，符号计算无法解题；(B) 数值计算能很快地求出该定积分。

(1) 符号计算解法

```
syms t x
ft = exp( - sin(t))
sx = int(ft,t,0,4)
ft =
1/exp(sin(t))
Warning: Explicit integral could not be found.
sx =
int(1/exp(sin(t)), t = 0..4)
```

(2) 数值计算解法

```
dt = 0.05;                          % 为在连续区间上取离散的计算点,先设定采样间隔
t = 0:dt:4;                         % 根据题意,时间区间的左端取为 0。右端点也必须确定,如取 4
                                    % 数值计算只能在有限的区间上,取有限个采样点
Ft = exp( - sin(t));                % 按函数关系,在有限个自变量采样点上,进行数值计算 f(t),
                                    % 并把计算得到的数值以 Ft 变量名加以保存
                                    % f(t)数据的总点数与 t 点数相同
Sx = dt * cumtrapz(Ft);             % 计算从 0 开始到每个采样点为止的区间内,Ft 曲线下的面积
                                    % 这个面积是由一个个宽度为 dt 的小矩形面积累加而得的
Sx(end)                             % 显示所求得的定积分值
plot(t,Ft,'*r','MarkerSize',4)      % 利用如图 3.1-2 所示图形,有助于了解所求原函数的性状
hold on
plot(t,Sx,'.k','MarkerSize',15)
hold off
xlabel('x')
legend('Ft','Sx')
ans =
    3.0632
```

说明

本例仅用来说明“符号计算解题能力有限”，说明学习和掌握数值计算方法之必要。

3.2 数值数组的创建和寻访

出于数值计算离散本质的考虑，也出于“向量化”快速处理数据的需要，MATLAB 总把数组看作存储和运算的基本单元，而标量数据被看作(1×1)的数组。因此理解和掌握数组的创建、寻访和计算就显得特别重要。本书将通过第 3.2 和第 3.3 节分别阐述数组的创建、寻访和运算。

3.2.1 一维数组的创建

就所创建一维数组的用途而言，大致分为两类：自变量数组和通用变量数组。

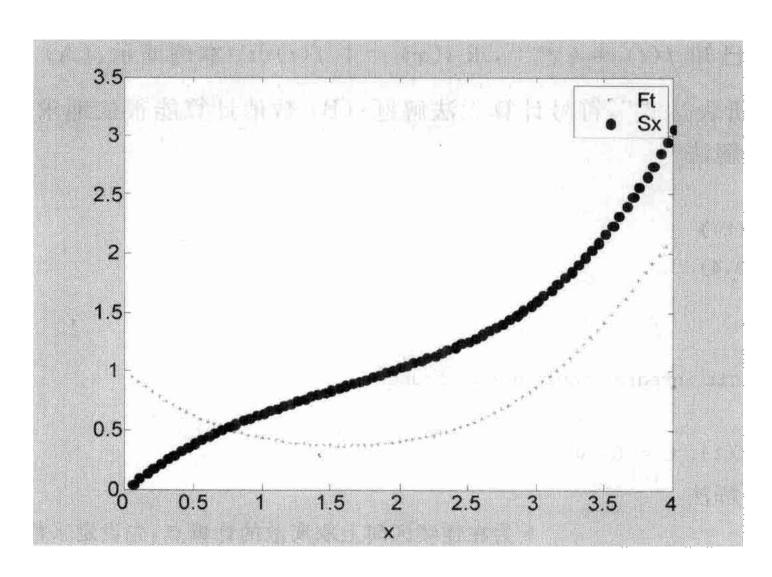


图 3.1-2 在区间[0, 4]中间的被积函数及其原函数的离散计算结果

1. 递增/减型一维数组的创建

这类数组的特点:数组元素值的大小按递增或递减的次序排列;数组元素值之间的"差"是"确定"的,即"等步长"的。这类数组主要用作函数的自变量(比如例 3.1-1 和例 3.1-2 中的数值自变量 t),for 循环中循环自变量等。

(1) "冒号"生成法

x = a : inc : b

说明

- a 是数组的第一个元素;inc 是采样点之间的间隔,即步长。若(b−a)是 inc 的整数倍,则所生成数组的最后一个元素等于 b,否则小于 b。
- a, inc, b 之间必须用冒号":"分隔。注意:该冒号必须在英文状态下产生。中文状态下的冒号将导致 MATLAB 操作错误!
- inc 可以省略。省略时,默认其取值为 1,即认为 inc=1。
- inc 可以取正数或负数。但要注意:inc 取正时,要保证 a<b;而 inc 取负时,要保证 a>b。

(2) 线性(或对数)定点法

x = linspace (a , b , n) 以 a, b 为左右端点,产生线性等间隔的($1\times n$)行数组

x = logspace (a , b , n) 以 a, b 为左右端点,产生对数等间隔的($1\times n$)行数组

说明

- n 是总采样点数,即一维数组的长度。
- x = linspace (a , b , n)等价于 $x=a:(b-a)/(n-1):b$。

2. 其他类型一维数组的创建

(1) 逐个元素输入法

这是最简单,但又最通用的构造方法。如 a0=[0.2,pi/2,−2,sin(pi/5),−exp(−3)]指

令就是一例。

(2) 运用 MATLAB 函数生成法

MATLAB 中有许多用来生成特殊形式数组的函数，如均匀分布随机数组的 rand(1,n)，全 1 数组 ones(1,n)等(参见表 3.2-1)。

例【3.2-1】 一维数组的常用创建方法举例。

```
a1 = 1:6                        % 缺省步长为 1
a2 = 0:pi/4:pi                  % 非整数步长
a3 = 1:-0.1:0                   % 负实数步长
a1 =
     1     2     3     4     5     6
a2 =
         0    0.7854    1.5708    2.3562    3.1416
a3 =
  Columns 1 through 6
    1.0000    0.9000    0.8000    0.7000    0.6000    0.5000
  Columns 7 through 11
    0.4000    0.3000    0.2000    0.1000         0

b1 = linspace(0,pi,4)           % 相当于 0:pi/3:pi
b2 = logspace(0,3,4)            % 创建数组[10^0  10^1  10^2  10^3]
b1 =
         0    1.0472    2.0944    3.1416
b2 =
           1          10         100        1000

c1 = [2  pi/2  sqrt(3)  3+5i]           % 采用逐个元素输入法构造数组
c1 =
  Columns 1 through 3
   2.0000              1.5708              1.7321
  Column 4
   3.0000 + 5.0000i

rand('twister',0)               % 为重现下面结果，伪随机发生器置 0 状态，详见例 4.3-4 说明。
c2 = rand(1,5)                  % 产生(1×5)的均布随机数组
c2 =
    0.5488    0.7152    0.6028    0.5449    0.4237
```

说明

- 以上演示产生的都是“行”数组。下面产生“列”数组的指令举例，请读者自己运行。

```
x1 = (1:6)' , x2 = linspace(0,pi,4)'
y1 = rand(5,1)
z1 = [2; pi/2; sqrt(3); 3+5i]
```

● 例中计算结果 a3，c1，c2 的显示格式与二个因素有关：一，Notebook 环境下，{Notebook\Notebok Options}菜单引出的 Notebook Options 对话窗中的 Format 选项。默认设置选项为 short。二，指令窗(Command Windows)的宽度。

3.2.2 二维数组的创建

除一维数组外，二维数组也是最常用的数组。

二维数组的手工输入方法已在第 1.3.3 节里做了初步介绍。本节将比较系统地叙述二维数组的几种创建方法。

1. 小规模数组的直接输入法

对于较小数组，从键盘上直接输入最简便。二维数组必须有以下三个要素：

● 整个输入数组必须以方括号“[ ]”为其首尾；

● 数组的行与行之间必须用分号“;”或回车键[Enter]隔离；

● 数组元素必须由逗号“,”或空格分隔。

例【3.2-2】 在 MATLAB 环境下，用下面三条指令创建二维数组 C。

```
a = 2.7358; b = 33/79;                        % 这两条指令分别给变量 a ,b 赋值。
C = [1,2 * a + i * b,b * sqrt(a);sin(pi/4),a + 5 * b,3.5 + i]
                                              % 这指令用于创建二维数组 C
C =
    1.0000              5.4716 + 0.4177i    0.6909
    0.7071              4.8244              3.5000 + 1.0000i
```

说明

● 分号“;”在“[ ]”方括号内时，它是数组行间的分隔符。

● 分号“;”用作为指令后的结束符时，将不在屏幕上显示该指令执行后的结果。

2. 中规模数组的数组编辑器创建法

当数组规模较大，元素数据比较冗长时，就不宜采用指令窗直接输入法，此时借助数组编辑器(见图 1.7-3)比较方便。下面举例说明具体创建方法。

例【3.2-3】 试用变量编辑器，把如下(3×6)的数组输入 MATLAB 内存，并命名为 A18。

```
0.6459    0.9637    0.5289    0.0710    0.8326    0.9786
0.4376    0.3834    0.5680    0.0871    0.7782    0.7992
0.8918    0.7917    0.9256    0.0202    0.8700    0.4615
```

(1) 单击工作空间浏览器(Workspace)中的图标，便在工作空间中引出一个名为 unnamed的变量。

(2) 双击 unnamed 变量引出一个与图 3.2-1 类似的空白界面。但数组中，除第一个元素为 0 外，其余均为“空白”。

(3) 在空白界面上，按照“行、列”次序输入数据。在最后一个数据 0.4615 输入结束后，必须按[Enter]键，或在数组编辑区内单击鼠标，使整个数组保存在 unnamed 变量中(见图 3.2-1)。

（4）在工作空间浏览器中，点中 unnamed 变量，利用弹出菜单的 {Rename}，把变量名修改成所需名称，比如 A18。

（5）假如该变量要供以后调用，建议再把此变量保存为 A18. mat 文件。

Variable Editor - A18

A18 <3x6 double>

	1	2	3	4	5	6	7
1	0.6459	0.9637	0.5289	0.0710	0.8326	0.9786	
2	0.4376	0.3834	0.5680	0.0871	0.7782	0.7992	
3	0.8918	0.7917	0.9256	0.0202	0.8700	0.4615	
4							

图 3.2-1　利用数组编辑器创建中规模数组

3. 中规模数组的 M 文件创建法

对于今后经常需要调用的数组，当数组规模较大而复杂时，为它专门建立一个 M 文件是值得的。下面通过一个简单例子来说明这种 M 文件的创建过程。

例【3.2-4】　为数组 AM 创建一个 MyMatrix. m 文件。以后每当需要 AM 数组时，只要运行 MyMatrix 文件，就可在内存生成 AM。

（1）打开文件编辑调试器，并在空白填写框中输入所需数组（见图 3.2-2）。

（2）最好，在文件的首行，编写文件名和简短说明，以便查阅（见图 3.2-2）。

（3）保存此文件，并且文件起名为 MyMatrix. m 。

（4）以后只要在 MATLAB 指令窗中，运行 MyMatrix. m 文件 ，数组 AM 就会自动生成于 MATLAB 内存中。

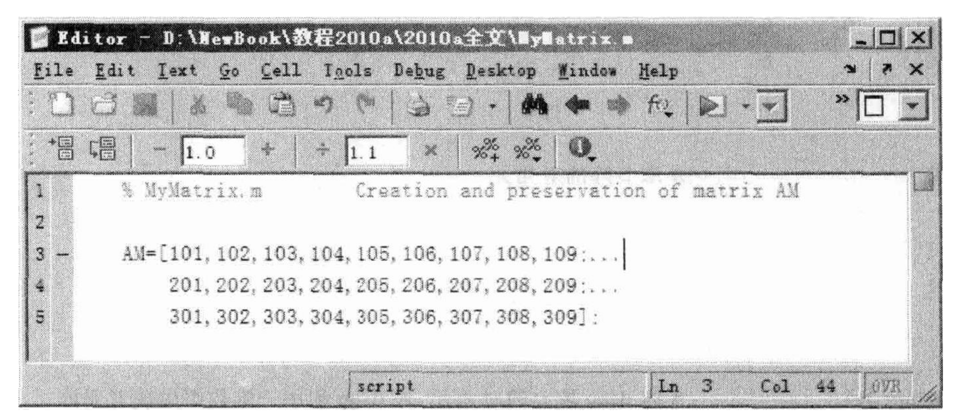


图 3.2-2　利用 M 文件创建数组

4. 利用 MATLAB 函数创建数组

在实际应用中，用户往往需要产生一些特殊形式的数组/矩阵。MATLAB 考虑到这方面的需要，提供许多生成特殊数组的函数。表 3.2-1 列出了最常用函数。

表 3.2-1 标准数组生成函数

指 令	含 义
diag	产生对角数组(对高维不适用)
eye	产生单位数组(对高维不适用)
magic	产生魔方数组(对高维不适用)
rand	产生均匀分布随机数组
randn	产生正态分布随机数组
ones	产生全 1 数组
zeros	产生全 0 数组
random	生成各种分布随机数组
randsrc	在指定字符集上生成均布随机数组
gallery	产生各种用途的测试数组/矩阵(参见第 4 章)

例【3.2-5】 标准数组产生的演示。

```
ones(2,4)                    % 产生(2×4)全 1 数组
ans =
     1     1     1     1
     1     1     1     1

randn('state',0)             % 把正态随机数发生器置 0
randn(2,3)                   % 产生(2×3)的正态随机阵
ans =
   -0.4326    0.1253   -1.1465
   -1.6656    0.2877    1.1909

D = eye(3)                   % 产生(3×3)的单位阵
D =
     1     0     0
     0     1     0
     0     0     1

diag(D)                      % 取 D 阵的对角元
ans =
     1
     1
     1

diag(diag(D))                % 内 diag 取 D 的对角元,外 diag 利用一维数组生成对角阵
ans =
     1     0     0
     0     1     0
     0     0     1
```

```
randsrc(3,20,[-3,-1,1,3],1)
                         % 在[-3,-1,1,3]字符集上产生(3×20)均布数组
                         % 随机发生器的状态设置为 1
ans =
    Columns 1 through 12
      -1    -1    -3     1    -3     1    -3     3     3    -3    -3     1
       1    -3    -1    -1     3    -1    -3    -1     3    -3    -1     1
      -3    -3    -1     1    -3     1     3     1    -3     3     3    -1
    Columns 13 through 20
       1     3    -1    -1    -1     1    -1    -3
       3     3     3     3    -3    -3    -3     1
      -3     1    -3    -1    -3    -1     1     1
```

3.2.3 二维数组元素的标识和寻访

二维数组元素及子数组的标识和寻访最具典型性。它既适用于一维数组，又不难推广到高维数组。对二维数组子数组进行标识和寻访的最常见格式见表 3.2-2。

表 3.2-2 子数组寻访和赋值格式汇总表

	格 式	使用说明
全下标法	A(r,c)	它由 A 的"r 指定行"和"c 指定列"上的元素组成
	A(r,:)	它由 A 的"r 指定行"和"全部列"上的元素组成
	A(:,c)	它由 A 的"全部行"和"c 指定列"上的元素组成
单下标法	A(:)	"单下标全元素"寻访，由 A 的各列按自左到右的次序，首尾相接而生成"一维长列"数组
	A(s)	"单下标"寻访，生成"s 指定的"一维数组，s 若是"行数组"（或"列数组"），则 $A(s)$ 就是长度相同的"行数组"（或"列数组"）
逻辑标识法	A(L)	"逻辑 1"寻访，生成"一维"列数组：由与 A 同样大小的"逻辑数组"L 中的"1"元素选出 A 的对应元素；按"单下标"次序排成长列组成

例【3.2-6】 本例演示：数组元素及子数组的各种标识和寻访格式；冒号的使用；end 的作用。

```
A = zeros(2,6)          % 创建(2×6)的全零数组
A(:) = 1:12             % 赋值号左边：单下标寻访(2×6)数组 A 的全部 12 个元素
                        % 赋值号右边：拥有 12 个元素的一位数组
                        % 注意"英文状态冒号:"的用法
A =
     0     0     0     0     0     0
     0     0     0     0     0     0
A =
     1     3     5     7     9    11
     2     4     6     8    10    12
```

```
A(2,4)                  % 双下标:数组 A 的第 2 行第 4 列元素
A(8)                    % 单下标:数组 A 的第 8 个元素
ans =
     8
ans =
     8

A(:,[1,3])              % 双下标:显示 A 的"第 1 和第 3 列上全部行的元素"
                        % 数组行标识位上,只有"英文状态冒号:",表示取"全部行"
A([1,2,5,6]')           % 单下标:把 A 数组第 1,2,5,6 个元素排成列向量
ans =
     1     5
     2     6
ans =
     1
     2
     5
     6


A(:,4:end)              % 双下标:显示 A 的"从第 4 列起到最后一列上全部行的元素"
                        % 在此,end 用于"列标识"。它表示"最后一列"
ans =
     7     9    11
     8    10    12

A(2,1:2:5) = [-1,-3,-5]            % 把右边 3 个数字分别赋向
                                   % A 数组第 2 行的第 1,3,5 个元素位置
                                   % 赋值时,要注意赋值号两边的元素数目相等
A =
     1     3     5     7     9    11
    -1     4    -3     8    -5    12

B = A([1,2,2,2],[1,3,5])    % 取 A 数组的 1,3,5 列的第 1 行元素作为 B 的第 1 行
                            % 取 A 数组的 1,3,5 列的第 2 行分别作为 B 的第 2,3,4 行
B =
     1     5     9
    -1    -3    -5
    -1    -3    -5
    -1    -3    -5


L = A<3                 % 赋值号右边:进行关系比较,产生逻辑结果。
```

```
                         % 产生与 A 维数大小相同的"0,1"逻辑数组:1 表示"真"
                         % 在此 L 数组中取 1 的位置对应的 A 数组元素小于 3
A(L) = NaN               % 把逻辑 1 标识位置上的元素赋为"非数"
L =
     1     0     0     0     0     0
     1     0     1     0     1     0
A =
   NaN     3     5     7     9    11
   NaN     4   NaN     8   NaN    12
```

3.2.4 数组操作技法综合

为了生成比较复杂的数组，或为了对已生成数组进行修改、扩展，MATLAB 提供了诸如反转、插入、提取、收缩、重组等操作。理解和掌握本节内容，对灵活使用 MATLAB 是重要的。最常用的操作函数见表 3.2－3。

表 3.2－3　数组操作函数

指　令	含　义
diag	提取对角元素，或生成对角阵
repmat	按指定的"行数、列数"铺放模块数组，以形成更大的数组
reshape	在总元素数不变的前提下，改变数组的"行数、列数"
flipud	以数组"水平中线"为对称轴，交换上下对称位置上的数组元素
fliplr	以数组"垂直中线"为对称轴，交换左右对称位置上的数组元素
rot90	把数组逆时针旋转 90°

例【3.2－7】 数组操作函数 reshape，diag，repmat 的用法；空阵[ ]删除子数组的用法。

```
a = 1:8                        % 产生(1×8)一维数组
A = reshape(a,4,2)             % 把一维数组 a 重排成(4×2)的二维数组
A = reshape(A,2,4)             % 再把(4×2)数组重组成(2×4)数组
a =
     1     2     3     4     5     6     7     8
A =
     1     5
     2     6
     3     7
     4     8
A =
     1     3     5     7
     2     4     6     8

b = diag(A)                 % 取(2×4)数组的对角元素形成(2×1)列数组
B = diag(b)                 % 据(2×1)列数组构造(2×2)对角阵
```

```
b =
     1
     4
B =
     1     0
     0     4
```

```
D1 = repmat(B,2,4)        % 把数组 B 当作模块,按(2×4)形式排放该模块,形成(4×8)数组
D1 =
     1     0     1     0     1     0     1     0
     0     4     0     4     0     4     0     4
     1     0     1     0     1     0     1     0
     0     4     0     4     0     4     0     4
```

```
D1([1,3],:) = [ ]        % 删除 D1 数组的第 1 和 3 行
D1 =
     0     4     0     4     0     4     0     4
     0     4     0     4     0     4     0     4
```

例【3.2-8】 函数 flipud,fliplr,rot90 对数组的操作体现着"矩阵变换"。

```
A = reshape(1:9,3,3)
A =
     1     4     7
     2     5     8
     3     6     9
```

B = flipud(A)　　% 上下对称交换,即意味着对 A 进行"行交换",即 $B=\begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}\cdot A$

```
B =
     3     6     9
     2     5     8
     1     4     7
```

C = fliplr(A)　　% 左右对称交换,意味着对 A 进行"列交换",即 $C=A\cdot\begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}$

```
C =
     7     4     1
     8     5     2
     9     6     3
```

```
D=rot90(A,2)        % 旋转 180°,意味着 D = [0 0 1; 0 1 0; 1 0 0]·A·[0 0 1; 0 1 0; 1 0 0]
D =
     9     6     3
     8     5     2
     7     4     1
```

3.3 数组运算

3.3.1 数组运算的由来和规则

1. 函数关系数值计算模型的分类

与符号计算不同,数值计算接受的是离散数据,在计算过程中的加减乘除等运算和函数运算是对离散数据集进行的,而最终的计算结果也是离散数据集。在数值计算实现的数学模型中,对离散数据进行处理的函数关系运算可归纳成如下三类。

(1) 个别的、无规律的数据集所执行函数关系运算

体现这种运算的程序通常是:不在循环体内的标量的表达式运算。

(2) 一组有规律数据需要反复执行的函数关系运算

这种运算的程序一般体现为:一个包含标量表达式计算的循环体。

(3) 一组有规律数据按照矩阵运算法则执行的运算

这种运算的程序实现一定是:包含标量表达式计算的一重或多重循环体。

2. 提高程序执行性能的三大措施

为了提高程序执行时的性能,MATLAB 针对三种不同类型的函数关系运算采取如下措施。

(1) 采用所谓的 JIT 加速器(JIT-Accelerator),提高 FOR 循环中标量函数关系运算的效率。

(2) 采用“数组运算”模式处理那些借助循环而反复执行的标量运算。这就是所谓的“向量化”运算。

(3) 采用“向量或矩阵运算”模式去执行“那些传统上靠多重循环标量运算完成的”矩阵计算。

其中第(2)和第(3)条措施,凸显出 MATLAB 面向数组/矩阵编程和运算的特点。这不仅使得 MATLAB 程序的书写有时与经典教科书的数学描述十分相近,而且大大提高了程序执行的速度。

3. 数组运算规则

(1) 两个同维同大小的($m\times n$)数组 $\boldsymbol{A}=[a_{ij}]_{m\times n}$ 和 $\boldsymbol{B}=[b_{ij}]_{m\times n}$ 的算术运算结果为数组

$\boldsymbol{C}=[c_{ij}]_{m\times n}$，这就意味着 $\boldsymbol{C}$ 数组的第(i,j)元素一定是数组 A 和 B 相同位置元素进行那指定算术运算的结果，即 $c_{ij}=a_{ij}\#b_{ij}$。在此，符号 # 可代表加减乘除幂运算中的任何一种运算。（程序算符见表 3.3-1。）

（2）设标量 a 和数组 $\boldsymbol{B}=[b_{ij}]_{m\times n}$ 进行算术运算的结果是 $\boldsymbol{C}=[c_{ij}]_{m\times n}$，这意味着 $c_{ij}=a\#b_{ij}$。（程序算符见表 3.3-1。）

（3）函数 $f(\cdot)$ 的数组运算规则是指“该函数对数组的逐个元素起作用”。该表述的数学含义是：对于$(m\times n)$数组 $\boldsymbol{X}=\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}=[x_{ij}]_{m\times n}$，定义 $f(\boldsymbol{X})=[f(x_{ij})]_{m\times n}$。

4. 数组运算符及数组运算函数

MATLAB 的数组运算和矩阵运算的运算符及其数学意义列于表 3.3-1 中，a_{ij} 和 b_{ij} 分别是数组（或矩阵）$\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的第(i,j)个元素；而服从数组运算规则的函数及其他算符列于表 3.3-2 中。

表 3.3-1 MATLAB 的数组/矩阵运算符及其数学意义

数组运算		矩阵运算	
数学模型描述	程序表达	数学模型描述	程序表达
A 的非共轭转置	A.'	**A** 的共轭转置	A'
$a_{ij}+b_{ij}$	A+B	$\boldsymbol{A}+\boldsymbol{B}$	A+B
$a_{ij}-b_{ij}$	A-B	$\boldsymbol{A}-\boldsymbol{B}$	A-B
$a_{ij}\times b_{ij}$	A.*B	$\boldsymbol{AB}$	A*B
a_{ij}/b_{ij} 或 $b_{ij}\backslash a_{ij}$	A./B 或 B.\A	$\boldsymbol{AB}^{-1}$ 或 $\boldsymbol{AB}^{+}$	A/B
		$\boldsymbol{B}^{-1}\boldsymbol{A}$ 或 $\boldsymbol{B}^{+}\boldsymbol{A}$	B\A
a_{ij}^b_{ij}	A.^B		
$a+b_{ij}$	a+B 或 a.+B	$a+b_{ij}$	a+B
$a-b_{ij}$	a-B 或 a.-B	$a-b_{ij}$	a-B
$a\times b_{ij}$	a.*B	$a\boldsymbol{B}$	a*B
a/b_{ij} 或 $b_{ij}\backslash a$	a./B 或 B.\a		
$a\backslash b_{ij}$ 或 b_{ij}/a	a.\B 或 B./a	$\frac{1}{a}\boldsymbol{B}$	B/a 或 a\B
$a^{b_{ij}}$	a.^B	（$\boldsymbol{B}$ 为方阵时）$a^{\boldsymbol{B}}$	a^B
b_{ij}^{a}	B.^a	（$\boldsymbol{B}$ 为方阵时）$\boldsymbol{B}^{a}$	B^a

说明

- 为了避免数组运算和矩阵运算的混淆，把两种运算符对照列出。
- 特别注意：数组运算程序表达算符中的“.”（英文状态下的小黑点）。
- 数组运算若在两个数组间进行，那么这两个数组必须维数大小相同。
- 凡 MATLAB 程序中出现 a+B，a−B 的形式，则总理解为“数组加、减”。换句话说，MATLAB 中，算符.＋等同于＋，算符.－等同于－。
- 矩阵除是 MATLAB 专门设计的一种运算。它有“左除\”和“右除/”之区别。矩阵除有丰富的内涵，详见第 4.2.3 节。

表 3.3-2　服从数组运算规则的函数及其他算符

		举　例
服从数组运算规则的函数	三角、反三角	sin，cos，tan，cot，sec，csc，asin，acos，atan，acos，asec，acsc
	双曲、反双曲	sinh，cosh，…，asinh，acosh，…
	指数、对数	exp，sqrt，pow2，log，log10，log2
	圆整、求余	ceil，floor，fix，round，mod，rem
	模、角、虚实部	abs，angle，real，imag，conj
	符号函数	sign
关系运算符		==，~=，>，<，>=，<=
逻辑运算符		&，|，~

3.3.2　数组运算和向量化编程

假如把标量看作“单件产品”，那么标量运算相当于“产品的单件生产”，这是效率低下的生产组织方式。把大量的“单件产品”组织在“流水线”加工，可以大大提高效率。这种思想在计算程序中的体现，就是“向量化编程”。在 MATLAB 中，若想达到向量化编程目的，就要尽量少地采用标量运算表达式，且尽可能使用数组/矩阵运算指令替代原先那些“包含标量运算表达式的循环体”。向量化程序不但可读性好，而且执行速度快。本节以算例形式展开。

例【3.3-1】　欧姆定律：$r=\frac{u}{i}$，其中 r,u,i 分别是电阻(Ω)、电压(V)、电流(A)。验证实验：据电阻两端施加的电压，测量电阻中流过的电流，然后据测得的电压、电流计算平均电阻值。（测得的电压电流具体数据见下列程序）。本例演示：数组运算符的作用；mean 指令的作用。

（1）非向量化程序

```
clear
vr = [0.89, 1.20, 3.09, 4.27, 3.62, 7.71, 8.99, 7.92, 9.70, 10.41];                    % 测量电压数据
ir = [0.028, 0.040, 0.100, 0.145, 0.118, 0.258, 0.299, 0.257, 0.308, 0.345]; % 测量电流数据
% ----------非向量化编程计算各数据点上的电阻值-----------
L = length(vr);                          % 数据点数目                                   <4>
for k = 1:L
    r(k) = vr(k)/ir(k);                  % 据各测量点的数据计算电阻
```

```
end                                                                    %      <7>
% --------------非向量化编程计算平均电阻值---------------
sr = 0;                                                                %      <8>
for k = 1:L
    sr = sr + r(k);                   %求所有测量点计算电阻之和(供计算电阻平均值用)
end
rm = sr/L                             %计算电阻平均值                         <12>
rm =
   30.5247
```

(2) 向量化程序

```
clear
vr = [0.89, 1.20, 3.09, 4.27, 3.62, 7.71, 8.99, 7.92, 9.70, 10.41]; %测量电压数据
ir = [0.028, 0.040, 0.100, 0.145, 0.118, 0.258, 0.299, 0.257, 0.308, 0.345]; %测量电流数据
r = vr./ir                            %计算各数据点上的电阻值                 <16>
rm = mean(r)                          %计算电阻平均值                         <17>
r =
   31.7857   30.0000   30.9000   29.4483   30.6780   29.8837   30.0669
30.8171   31.4935   30.1739
rm =
   30.5247
```

说明

- 在非向量化编程中,第〈4〉到第〈7〉条指令,借助循环计算各数据点上的电阻。而在采用"数组除"运算后,整个过程仅用一条指令〈16〉就完成。
- 在非向量化编程中,第〈8〉到第〈12〉条指令,借助循环计算各数据点电阻的平均值。可是在向量化程序中,只要一条指令〈17〉就可算得平均电阻。
- 在编程中,要尽量避免使用循环,特别是 while 环。因为在循环中,内存不断重新分配,使计算速度变慢。
- 只要知道"数组除"的含义和算符,用户不难写出第〈16〉条指令。需要注意的是,用户应该克服其他语言中养成的编程习惯,而不断熟悉"数组运算符"。
- 至于第〈17〉条指令的使用,要求用户理解 MATLAB 函数 mean 的意义和用法。应该指出:假如用户能在所编程序中,尽可能多地采用 MATLAB 现成的函数,那么你所编程序的质量就可能比较高,即其可靠、快速和可读性较好。

例【3.3-2】 用间距为 0.1 的水平线和垂直线均匀分割 $x\in[-5,5]$,$y\in[-2.5,2.5]$的矩形域,在所有水平线和垂直线交点上计算函数 $z=\sin|xy|$ 的值,并图示。本例演示:服从数组运算规则的函数的作用;向量化编程;如何判断两个"二维双精度数组"是否相等;绘制二元函数图形的基本原理。

(1) 非向量化编程

```
clear
x = -5:0.1:5;                         % x轴上的方格格点刻度
y = (-2.5:0.1:2.5)';                  % y轴上的方格格点刻度
```

```
N = length(x);                    % x 轴上的格点总数
M = length(y);                    % y 轴上的格点总数
for ii = 1:M
    for jj = 1:N
        X0(ii,jj) = x(jj);        % 指定矩形域内所有格点的 x 坐标
        Y0(ii,jj) = y(ii);        % 指定矩形域内所有格点的 y 坐标
        Z0(ii,jj) = sin(abs(x(jj) * y(ii)));
                                  % 矩形域所有格点坐标(x, y)对应的函数值
    end
end
```

(2) 向量化编程

```
[X,Y] = meshgrid(x,y);            % 指定矩形域内所有格点的(x, y)坐标
Z = sin(abs(X. * Y));             % 数组运算计算矩形域所有格点坐标(x, y)对应的函数值
```

(3) 比较二维双精度数组是否相等

```
norm(Z - Z0)                      % 若此范数值接近 eps,那么认为相等
ans =
     0
```

(4) 在指定矩形域上绘制二元函数图形(见图 3.3-1)

```
surf(X,Y,Z)
xlabel('x')
ylabel('y')
shading interp
view([190,70])
```

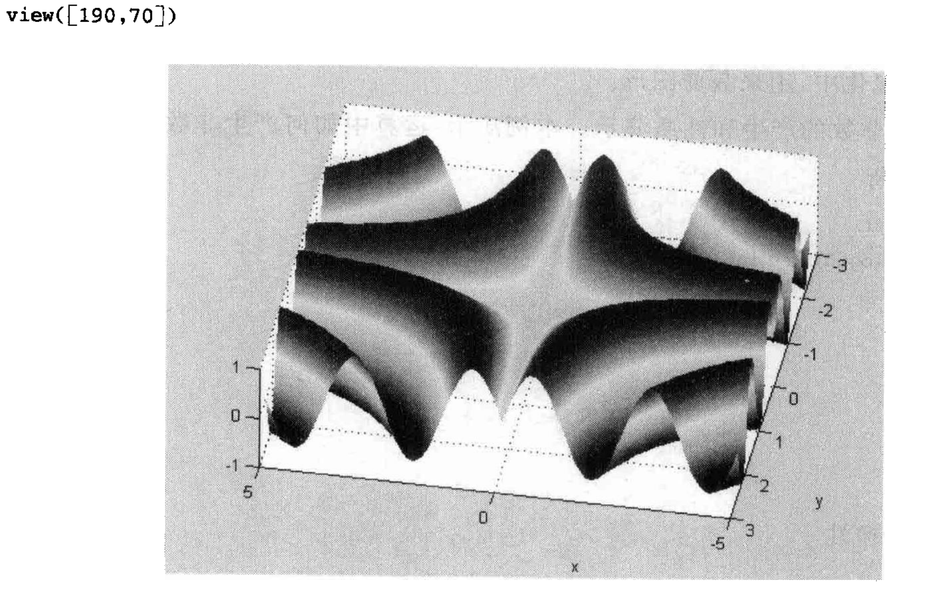


图 3.3-1　指定域上的二元函数图形

说明

- 若采用非向量化编程，曲面点(x, y, z)坐标的计算需采用二重循环进行。这是效率低下的做法。
- 向量化编程要求编程者熟悉数组运算和 MATLAB 的指令。

3.4 “非数”和“空”数组

这是 MATLAB 中特有的两个概念和“预定义变量”。

3.4.1 非　数

(1)非数的产生和性质

按 IEEE 规定，$\left(\frac{0}{0}\right)$，$\left(\frac{\infty}{\infty}\right)$，$(0\times\infty)$，$(\infty-\infty)$等运算都会产生非数(Not a Number)。该非数在 MATLAB 中用 NaN 或 nan 记述。

根据 IEEE 数学规范，NaN 具有以下性质：

- NaN 参与运算所得的结果也是 NaN，即具有传递性。
- 非数没有“大小”概念，因此不能比较两个非数的大小。

(2) 非数的功用

- 真实记述$\left(\frac{0}{0}\right)$，$\left(\frac{\infty}{\infty}\right)$，$(0\times\infty)$，$(\infty-\infty)$运算的后果。
- 避免可能因$\left(\frac{0}{0}\right)$，$\left(\frac{\infty}{\infty}\right)$，$(0\times\infty)$，$(\infty-\infty)$运算而造成程序执行的中断。
- 在测量数据处理中，可以用来标识“野点(非正常点)”。
- 在数据可视化中，用来裁剪图形。

例【3.4-1】 非数的产生和性质演示。本例演示：运算中如何产生非数；在运算过程中的传递；非数如何判断。

(1) 非数的产生

```
a = 0/0,b = 0 * log(0),c = inf - inf
a =
   NaN
b =
   NaN
c =
   NaN
```

(2) 非数的传递性

```
0 * a,sin(a)
ans =
   NaN
ans =
   NaN
```

（3）非数的属性判断

```
class(a)            % 数据类型归属
isnan(a)            % 该指令是唯一能正确判断非数的指令。
ans =
double
ans =
     1
```

【3.4－2】 非数元素的寻访。本例演示：指令 isnan，find 的用法。

```
% 创建带非数的二维数组
rand('twister',0)       % 将随机发生器置 0。目的是使读者便于把自己运算结果与本书对照。
R = rand(2,5);R(1,5) = NaN;R(2,3) = NaN
R =
    0.5488    0.6028    0.4237    0.4376       NaN
    0.7152    0.5449       NaN    0.8918    0.3834

LR = isnan(R)                 % 对数组元素是否非数进行判断                              <3>
LR =
     0     0     0     0     1
     0     0     1     0     0

% 找出非数元素的位置标识
si = find(LR)                              % 确定非数的“单下标”标识
[ri,ci] = ind2sub(size(R),si)              % 把“单下标”转换成“全下标”标识
[rj,cj] = find(LR)           % 直接确定非数的“全下标”。结果应与上一条指令相同
disp('非数在二维数组 R 中的位置')
disp(['单下标时的第',int2str(si(1)),'和第',int2str(si(2)),'个元素'])
si =
     6
     9
ri =
     2
     1
ci =
     3
     5
rj =
     2
     1
cj =
     3
     5
```

```
非数在二维数组 R 中的位置
单下标时的第 6 和第 9 个元素
```

说明

为确定非数的位置只能用指令〈3〉进行，而不能用 R==NaN。

3.4.2 “空”数组

“空”数组是 MATLAB 为操作和表述需要而专门设计的一种数组。二维“空”数组，用一对方括号表示。至于其他高维数组，只要数组某维长度为 0 或若干维长度均为 0，则该数组就是“空”数组。

“空”数组的功用：(1) 在没有“空”数组参与运算时，计算结果中的“空”可以合理地解释“所得结果的含义”；(2) 运用“空”数组对其他非空数组赋值，可以使数组变小，但不能改变那数组的维数。

说明

- 不要把“空”数组与全零数组混淆。这是两个不同的概念。
- 不要把“空”数组看成“虚无”，它确实地存在。利用 which，who，whos，以及变量浏览器都可以验证它的存在。
- 唯一能正确判断一个数组是否“空”数组的指令是 isempty。
- “空”数组在运算中不具备传递性。对运算中出现的“空”结果，解释要谨慎。

例【3.4-3】 关于“空”数组的算例。

(1) 创建“空”数组的几种方法

```
a = []
b = ones(2,0),c = zeros(2,0),d = eye(2,0)
f = rand(2,3,0,4)
a =
     []
b =
   Empty matrix: 2 - by - 0
c =
   Empty matrix: 2 - by - 0
d =
   Empty matrix: 2 - by - 0
f =
   Empty array: 2 - by - 3 - by - 0 - by - 4
```

(2) “空”数组的属性

```
class(a)                  % “空”的数据类别
isnumeric(a)              % 是数值数组类吗
isempty(a)                % 唯一可正确判断数组是否“空”的指令
ans =
double
ans =
```

```
     1
ans =
     1

which a                 % 变量 a 是什么
ndims(a)                % 数组 a 的维数
size(a)                 % a 数组的大小
a is a variable.
ans =
     2
ans =
     0     0
```

(3)"空"数组用于子数组的删除和大数组的大小收缩

```
A = reshape(-4:5,2,5)
A =
    -4    -2     0     2     4
    -3    -1     1     3     5

A(:,[2,4]) = []                 % 删除 A 的第二、四列
A =
    -4     0     4
    -3     1     5
```

3.5 关系操作和逻辑操作

在程序流控制中,在逻辑、模糊逻辑推理中,都需要对一类是非问题作出"是真,是假"的回答。为此,MATLAB 设计了关系操作、逻辑操作和一些相关函数。虽在其他程序语言中也有类似的关系、逻辑运算,但 MATLAB 作为一种比较完善的科学计算环境,有其自身的特点。

MATLAB 约定:

- 在所有关系表达式和逻辑表达式中,作为输入的任何非 0 数都被看作是"逻辑真",而只有 0 才被认为是"逻辑假"。
- 所有关系表达式和逻辑表达式的计算结果,即输出,是一个由 0 和 1 组成的"逻辑数组(Logical Array)"。在此数组中的 1 表示"真",0 表示"假"。
- 逻辑数组是一种特殊的数值数组。既然它归属"数值类",那么与"数值类"有关的操作和函数对它也同样适用。但它又不同于普通的数值,它还表示着对事物的判断结论"真"与"假"。因此,它有自身的特殊用途,如数组寻访等。

3.5.1 关系操作

关系操作符的指令及其含义列于表 3.5-1 中。

表 3.5-1　关系操作符

指　令	含　义	指　令	含　义
<	小于	>=	大于等于
<=	小于等于	==	等于
>	大于	~=	不等于

说明

- 标量可以与任何维数组进行比较。比较在此标量与数组每个元素之间进行，因此比较结果将与被比数组同维。
- 当比较量中没有标量时，关系符两端进行比较的数组必须维数相同。比较在两数组相同位置上的元素间进行，因此比较结果将与被比数组同维。

例【3.5-1】　关系运算示例。本例演示：标量与数组的关系运算；数组与数组的关系运算。

```
A = 1:9,B = 10 - A
r0 = (A<4)              % 给出"对 A 数组每个元素是否小于 4"的情况判断
r1 = (A = = B)          % 给出"A, B 两数组对应元素是否相等"的情况判断
A =
     1     2     3     4     5     6     7     8     9
B =
     9     8     7     6     5     4     3     2     1
r0 =
     1     1     1     0     0     0     0     0     0
r1 =
     0     0     0     0     1     0     0     0     0
```

例【3.5-2】　关系运算应用。本例演示：利用关系运算认定元素为 0 的位置；利用 eps 求近似极限的处理方法；y 数组中的非数，在图形中表现为"缺口"（见图 3.5-1）；采用数组运算编写程序。

```
t = - 3 * pi:pi/10:3 * pi;     % 该自变量数组中，存在 0 值
y = sin(t)./t;                 % 在 t = 0 处，按 IEEE 规则，计算将产生非数
tt = t + (t == 0) * eps;       % 逻辑数组参与运算，使 0 元素被一个"机器零"小数代替
yy = sin(tt)./tt;              % 用数值可算的 sin(eps)/eps 近似替代 sin(0)/0 极限
subplot(1,2,1),plot(t,y),axis([ - 9,9, - 0.5,1.2]),
xlabel('t'),ylabel('y'),title('残缺图形')
subplot(1,2,2),plot(tt,yy),axis([ - 9,9, - 0.5,1.2])
xlabel('tt'),ylabel('yy'),title('正确图形')
```

说明

- 本例近似处理极限的方法具有特殊性。关于"近似数值极限"的使用，请参见4.1.1 节。
- 本例也演示了非数在绘图中的一个用途：能对曲线或曲面进行剪裁。

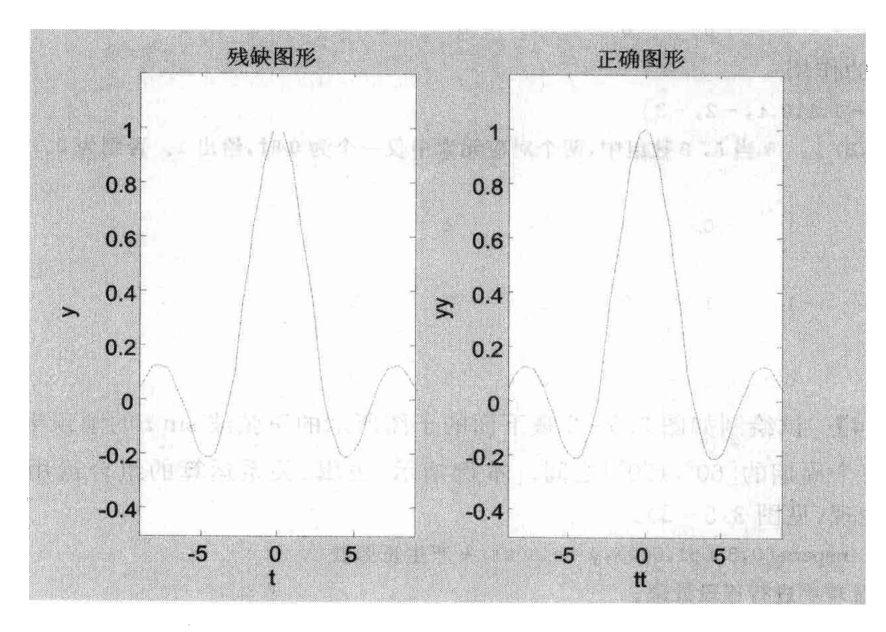


图 3.5-1　采用近似极限处理前后的图形对照

3.5.2　逻辑操作

上节仅介绍了简单关系操作。逻辑操作的引入，将使复杂关系运算成为可能。逻辑运算的操作符列于表 3.5-2 中。

表 3.5-2　逻辑操作符

指　令	&	|	~	xor
含　义	与	或	非	与非

说明

- 标量可以与任何维数组进行逻辑运算。运算比较在标量与数组每个元素间进行，因此运算结果与参与运算的数组同维。
- 当逻辑运算中没有标量时，参与运算的数组必须维数相同。运算在两数组相同位置上的元素间进行，因此运算结果数组必定和参与运算的数组同维。

例【3.5-3】　逻辑操作和关系操作。本例演示：逻辑、关系操作的组合；xor 的功能。

(1) 逻辑、关系操作的组合

```
A=[-2,-1,0,0,1,2,3]
L1=~(A>1)                   %判断A中,哪些元素不大于1
L2=(A>0)&(A<2)              %判断A中,哪些元素大于0且小于3
A =
     -2     -1      0      0      1      2      3
L1 =
      1      1      1      1      1      0      0
L2 =
```

```
        0     0     0     0     1     0     0
```

(2) xor 的作用

```
A,B=[0,-1,1,0,1,-2,-3]
C=xor(A,B)      % 当 A, B 数组中,两个对应元素中仅一个为 0 时,给出 1。否则为 0。
A =
      -2    -1     0     0     1     2     3
B =
       0    -1     1     0     1    -2    -3
C =
       1     0     1     0     0     0     0
```

例【3.5-4】 试绘制如图 3.5-2 最下面的子图所示的正弦波 sin t 的削顶半波整流波形，削顶发生在每个周期的[60°,120°]之间。本例演示：逻辑、关系运算的组合运用；逐段解析函数的计算和表现（见图 3.5-2）。

```
clear,t=linspace(0,3*pi,500);y=sin(t);% 产生正弦波
 % 从自变量着手进行逐段处理。
 z1=((t<pi)|(t>2*pi)).*y;                              % 获得整流半波              <3>
 w=(t>pi/3&t<2*pi/3)+(t>7*pi/3&t<8*pi/3);  % 关系逻辑运算和数值运算              <4>
 wn=~w;                                                                  %        <5>
 z2=w*sin(pi/3)+wn.*z1;                                  % 获得削顶整流半波          <6>
 subplot(4,1,1),plot(t,y,':r'),axis([0,10,-1.5,1.5])
 ylabel('y'),grid on
 subplot(4,1,2),plot(t,z1,':r'),axis([0,10,-0.2,1.5]),ylabel('z1')
 subplot(4,1,3),plot(t,wn,':r'),axis([0,10,-0.2,1.5]),ylabel('wn')
 subplot(4,1,4),plot(t,z2,'-b'),axis([0,10,-0.2,1.5]),ylabel('z2')
```

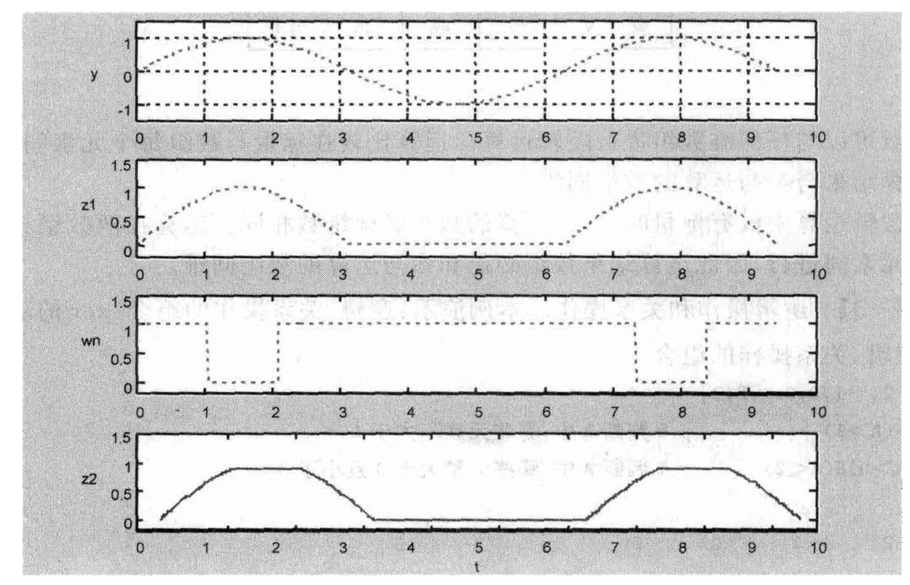


图 3.5-2 逐段解析函数的产生

```
xlabel('t')
```

说明

- 指令〈3〉使$(\pi,2\pi)$时间范围内的函数值为 0,削去负半波。
- 指令〈4〉产生的 w 向量中,除对应时间$\left(\frac{\pi}{3},\frac{2\pi}{3}\right)$,$\left(\frac{7\pi}{3},\frac{8\pi}{3}\right)$区间的采样点取 1 外,其余均取 0。而指令〈5〉产生的 wn 向量中元素的取值恰与 w 向量相反。
- 指令〈3〉,〈4〉是利用逻辑数组进行数值运算的范例。从本例可以看到:MATLAB 采用"0,1"表示"真、假"的措施,大大开拓了数值运算的应用范围。

3.5.3　常用逻辑函数

MATLAB 中能给出"逻辑数组"类型计算结果的函数有很多。它们包括:含 0 元素数组判断函数、逻辑数组创建函数(从 MATLAB 7.0 版本起配置)、数据对象判断函数、数据类型判断函数等。详见表 3.5-3。

表 3.5-3　常用逻辑函数

分　类	具体描述			
含 0 数组判断	all	数组 **A** 不含 0 元素,返回 1		
	any	数组 **A** 不是全 0 元素,返回 1		
生成逻辑数组	false	按指定大小,创建全 0 逻辑数组		
	true	按指定大小,创建全 1 逻辑数组		
	logical	创建逻辑数组:1 对应输入数组中的非 0 元素,其余都为 0		
数据对象判断	isempty	是否空阵	isprime	是否质数
	isfinite	是否有限数	isreal	是否实数
	isinf	是否无穷大	isletter	是否字母(用于字符串)
	isnan	是否非数	isspace	是否空格(用于字符串)
数据类型判断	isa	是否指定类别	ishandle	是否图柄
	ischar	是否字符串	islogical	是否逻辑类型
	isglobal	是否全局变量	isnumeric	是否数值类型

习题 3

1. 要求在闭区间$[0,2\pi]$上产生具有 10 个等距采样点的一维数组。试用两种不同的指令实现。(提示:冒号生成法,定点生成法)
2. 由指令 rand('twister',0),A=rand(3,5)生成二维数组 A,试求该数组中所有大于 0.5 的元素的位置,分别求出它们的"全下标"和"单下标"。(提示:find 和 sub2ind)
3. 在使用 123 作为 rand 随机数发生器的初始化状态的情况下,写出产生长度为 1000 的"等概率双位(即取-1,+1)取值的随机码"程序指令,并给出-1 码的数目。(提示:rand, randn, randsrc 等都可以用来产生所需码。注意:"关系符=="、"求和指令 sum"的应用。)
4. 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$,运行指令 B1=A.^(0.5), B2=A^(0.5),可以观察到不同运算

方法所得结果不同。(1) 请分别写出根据 B1，B2 恢复原矩阵 **A** 的程序。(2) 用指令检验所得的两个恢复矩阵是否相等。(提示：数组乘、矩阵乘。注意：范数指令 norm 用途。)

5. 在时间区间[0,10]中，绘制 $y=1-e^{-0.5t}\cos 2t$ 曲线。要求分别采取标量循环运算法和数组运算法编写两段程序绘图。(注意：体验数组运算的简捷。)
6. 先运行 clear,format long,rand('twister',1),A=rand(3,3)，然后根据 **A** 写出两个矩阵：一个对角阵 **B**，其相应元素由 **A** 的对角元素构成；另一个矩阵 **C**，其对角元素全为 0，而其余元素与对应的 **A** 阵元素相同。(提示：diag)
7. 先运行指令 x=-3*pi:pi/15:3*pi; y=x; [X,Y]=meshgrid(x,y); warning off; Z=sin(X).*sin(Y)./X./Y; 产生矩阵 **Z**。(1) 矩阵 **Z** 中有多少个"非数"数据？(2) 用指令 surf(X,Y,Z); shading interp 观察所绘的图形；(3) 请写出绘制相应的"无裂缝"图形的全部指令。(提示：isnan，sum，eps)
8. 下面有一段程序，企图用来解决如下计算任务：有矩阵 $\boldsymbol{A}_k=\begin{bmatrix}1 & k+1 & \cdots & 9k+1\\ 2 & k+2 & \cdots & 9k+2\\ \vdots & \vdots & & \vdots\\ k & 2k & \cdots & 10k\end{bmatrix}$，

当 k 依次取 10，9，8，7，6，5，4，3，2，1 时，计算矩阵 $\boldsymbol{A}_k$"各列元素的和"，并把此求和结果存放为矩阵 **Sa** 的第 k 行。例如 $k=3$ 时，**A** 阵为 $\begin{bmatrix}1 & 4 & \cdots & 28\\ 2 & 5 & \cdots & 29\\ 3 & 6 & \cdots & 30\end{bmatrix}$，此时它各列元素的和是一个(1×10)行数组[6 15 … 87]，并把它保存为 **Sa** 的第 3 行。问题：该段程序的计算结果对吗？假如计算结果不正确，请指出错误发生的根源，并改正之。

```
for k = 10: - 1:1
    A = reshape(1:10 * k,k,10);
    Sa(k,:) = sum(A);
end
Sa
```

(提示：本题专为揭示 sum 对行数组的作用而设计。仔细观察以下程序运行后所得到的 Sa 正确吗？ for k=10:-1:1; A=reshape(1:10*k,k,10); Sa(k,:)=sum(A); end; Sa。)

第4章 数值计算

与符号计算相比，数值计算在科研和工程中的应用更为广泛。MATLAB 也正是凭借其卓越的数值计算能力而著称。随着科研领域、工程实践的数字化进程的深入，具有数字化本质的数值计算就显得日益重要。

较之 10 年、20 年前，在计算机软硬件的支持下，当今人们所能拥有的计算能力已经得到了巨大的提升。这就自然地激发了人们从新的计算能力出发去学习、理解概念的欲望，鼓舞了人们用新的计算能力试探解决实际问题的雄心。

鉴于当今高校本科教学偏重符号计算和便于手算简单示例的实际，也出于帮助读者克服对数值计算生疏感的考虑，本章在内容安排上仍从微积分开始。这一方面与第 2 章符号计算相呼应，另一方面通过微积分说明数值计算离散本质的微观和宏观影响。

为便于读者学习，本章内容的展开脉络基本上沿袭高校数学教程，而内容深度力求控制在高校本科水平。考虑到知识的跳跃和交叉，本书对重要概念、算式、指令进行了尽可能完整地说明。

4.1 数值微积分

4.1.1 近似数值极限及导数

在 MATLAB 数值计算中，没有专门求极限和导数的指令。原因是：在数值浮点体系中，由于数值精度有限，不能表示无穷小量，不能准确描述一个数的邻域。但这并不意味着数值计算不能应用于导数等函数邻域概念有关的问题。事实上，数值计算是解各类微分方程的最主要工具。

编写本节的目的是：(1) 提醒读者高度重视有限精度浮点表示的离散本质，不要贸然自行编写数值计算程序进行求极限和导数的运算；(2) 在数值近似导数非求不可的情况下，自变量的增量选取一定要大于原数据相对精度的 10 倍以上；(3) 在解算极值、积分、微分方程等数值问题时，要尽量使用 MATLAB 提供的现成指令，严格遵循指令使用规则和仔细理解相关说明。

在 MATLAB 数值计算中，既没有专门的求极限指令，也没有专门的求导指令。但 MATLAB提供了与“求导”概念有关的“求差分”指令：

dx=diff(X)	求差分
FX = gradient(F)	求一元(函数)梯度
[FX,FY] = gradient(F)	求二元(函数)梯度

说明

- 对 diff 而言，当 X 是向量时，dx=X(2:n) －X(1:n－1)；当 X 是矩阵时，dx=X(2:n,

:)－X(1:n－1，:)。注意:dx 的长度比 X 的长度短少一个元素。

- 对 gradient 而言，当 F 是向量时，FX(1)＝F(2)－F(1)，FX(end)＝F(end)－F(end－1)，FX(2:end－1)＝(F(3:end)－F(1:end－2))/2。梯度采用“内点中心差分”计算。注意:Fx 的长度与 F 相同。
- 当在 gradient 中的 F 是矩阵时，FX，FY 是与 F 同样大小的矩阵。FX 的每行给出 F 相应行元素间的“梯度”；FY 的每列给出 F 相应列元素间的“梯度”。

例【4.1-1】 设 $f_1(x)=\dfrac{1-\cos 2x}{x\sin x}$，$f_2(x)=\dfrac{\sin x}{x}$，试用机器零阈值 eps 替代理论 0 计算极限 $L_1(0)=\lim\limits_{x\to 0}f_1(x)$，$L_2(0)=\lim\limits_{x\to 0}f_2(x)$。本例演示：除非数值近似法求的极限经过理论验证，否则绝不要借助数值法求取极限。

```
% 不可信的“极限的数值近似计算”
x = eps;
L1 = (1 - cos(2 * x))/(x * sin(x)),        % 将得到一个错误的极限值
L2 = sin(x)/x,                             % 恰巧与理论值一致
L1  =
     0
L2  =
     1

% 可信的“极限的符号计算”
syms t
f1 = (1 - cos(2 * t))/(t * sin(t));
f2 = sin(t)/t;
Ls1 = limit(f1,t,0)
Ls2 = limit(f2,t,0)
Ls1  =
2
Ls2  =
1
```

说明

- 理论分析表明 $f_1(x)=2f_2(x)$，而且 $L_2(0)=\lim\limits_{x\to 0}f_2(x)=\lim\limits_{x\to 0}\dfrac{\sin x}{x}=1$。
- 借助符号计算所求的极限与理论值一致。
- 而用数值法近似计算的极限与理论不一致。在此，再次提醒读者：不要企图借助数值计算求取极限。

例【4.1-2】 已知 $x=\sin(t)$，求该函数在区间$[0,2\pi]$中的近似导函数。本例演示：自变量增量的适当取值对数值导函数的精度影响极大。

(1) 计算数值导数时，自变量的增量取得过小(在 eps 数量级)(见图 4.1-1)

```
d = pi/100;
t = 0:d:2 * pi;
x = sin(t);
```

```
dt = 5 * eps;                     % 增量为 eps 数量级
x_eps = sin(t + dt);
dxdt_eps = (x_eps - x)/dt;        % 以 dt = 5 * eps 为增量算得的数值导数
plot(t,x,'LineWidth',5)
hold on
plot(t,dxdt_eps)
hold off
legend('x(t)','dx/dt')
xlabel('t')
```

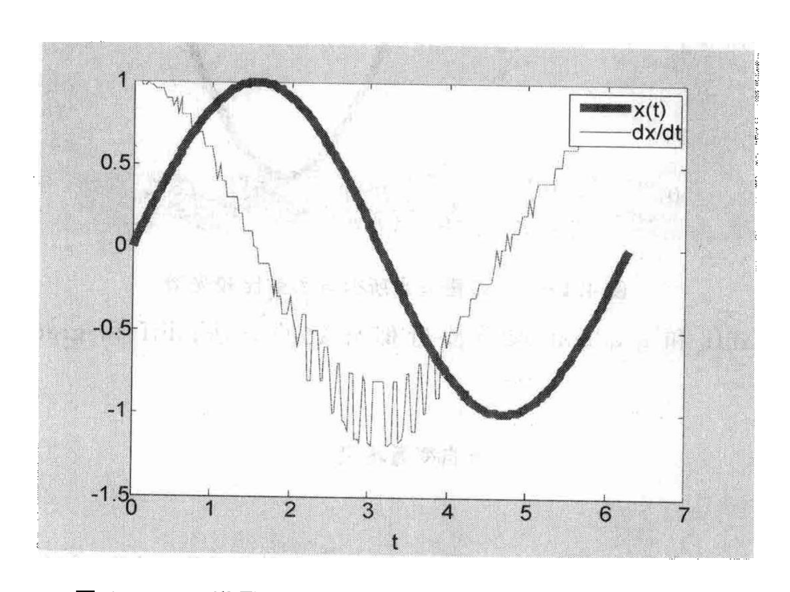


图 4.1-1　增量过小引起有效数字严重丢失后的毛刺曲线

（2）计算数值导数时，自变量的增量取得适当（见图 4.1-2）

```
x_d = sin(t + d);
dxdt_d = (x_d - x)/d;             % 以 d = pi/100 为增量算得的数值导数
plot(t,x,'LineWidth',5)
hold on
plot(t,dxdt_d)
hold off
legend('x(t)','dx/dt')
xlabel('t')
```

说明

- 本例说明：即使被导函数数据是从双精度计算获得的，数值导数仍然受计算中的有限精度困扰。当自变量增量 dt 取得太小时，$f(t+dt)$ 与 $f(t)$ 的数值十分接近，它们的高位有效数字完全相同。这样计算 $df=f(t+dt)-f(t)$ 时，$f(t+dt)$ 和 $f(t)$ 相减造成 dt 的许多高位有效数字消失，导致精度急剧变差。
- 再次强调：数值导数的使用应十分谨慎。

例【4.1-3】　已知 $x=\sin(t)$，采用 diff 和 gradient 计算该函数在区间$[0,2\pi]$中的近似导

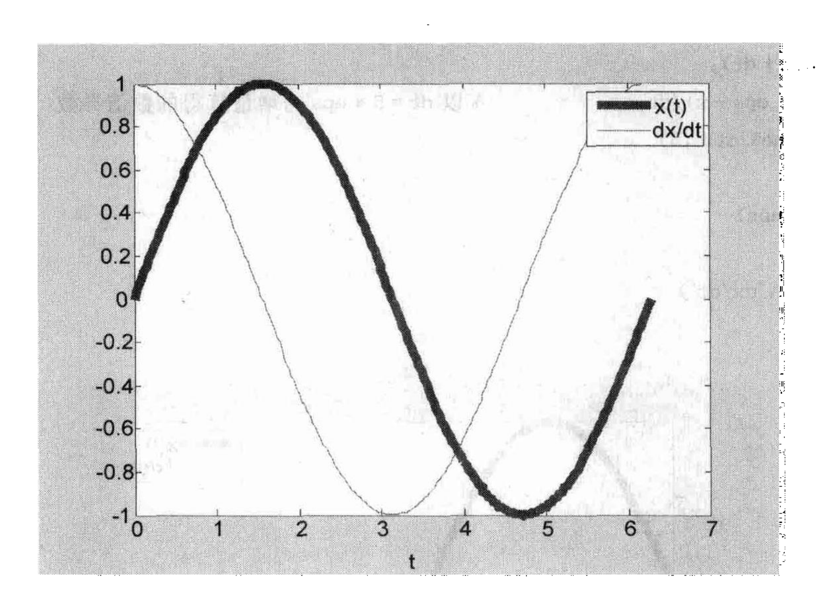


图 4.1-2 增量适当所得导函数比较光滑

函数。本例演示：diff 和 gradient 求数值近似导数的方法；diff 和 gradient 求导的差别（见图 4.1-3）。

```
clf
d=pi/100;                              % 自变量增量
t=0:d:2*pi;
x=sin(t);
dxdt diff=diff(x)/d;                   % diff 求得的近似数值导数。注意：除以 d
dxdt grad=gradient(x)/d;               % gradient 求得的近似数值导数。注意：除以 d
subplot(1,2,1)
plot(t,x,'b')
hold on
plot(t,dxdt grad,'m','LineWidth',8)
plot(t(1:end-1),dxdt diff,'.k','MarkerSize',8)
axis([0,2*pi,-1.1,1.1])
title('[0, 2\pi]')
legend('x(t)','dxdt {grad}','dxdt {diff}','Location','North')
xlabel('t'),box off
hold off
subplot(1,2,2)
kk=(length(t)-10):length(t); % t 数组中最后 11 个数据的下标
hold on
plot(t(kk),dxdt grad(kk),'om','MarkerSize',8)
plot(t(kk-1),dxdt diff(kk-1),'.k','MarkerSize',8)
title('[end-10, end]')
legend('dxdt {grad}','dxdt {diff}','Location','SouthEast')
xlabel('t'),box off
```

```
hold off
```

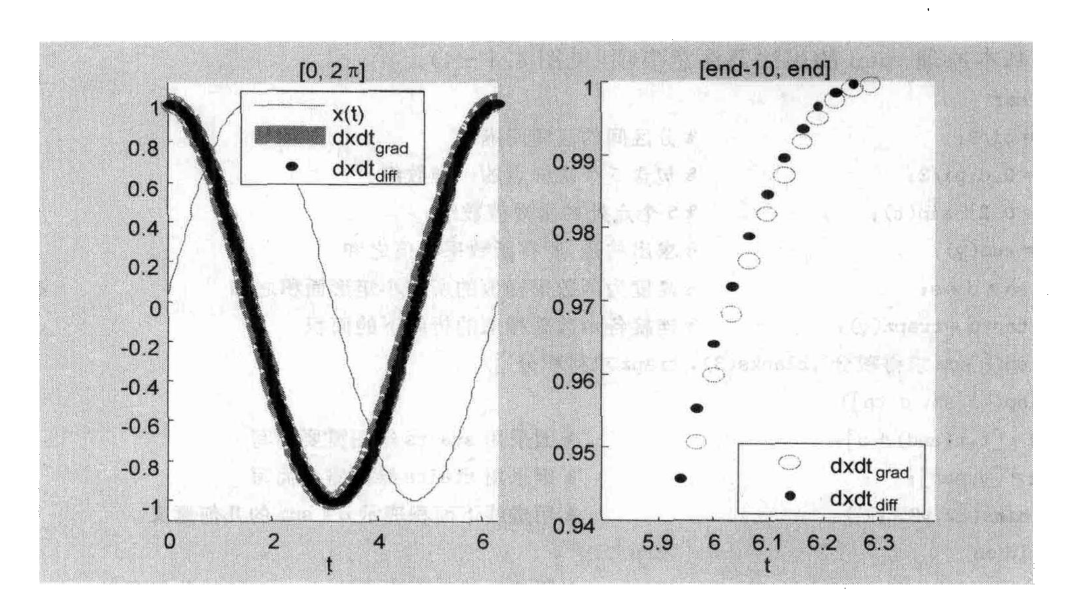


图 4.1-3 diff 和 gradient 求数值近似导数的异同比较

说明

- 图 4.1-3 中左图表明，从宏观看，diff 和 gradient 所求得近似导数大体相同。
- 从图 4.1-3 中右图可以看到，diff 和 gradient 不仅在数值上有差异，而且 diff 没有给出最后一点的导数。

4.1.2 数值求和与近似数值积分

Sx=sum(X)	沿列方向求和 $Sx(k)=\sum_{i=1}^{m} X_{m\times n}(i,k)$
Scs=cumsum(X)	沿列方向求累计和
St=trapz(x, y)	采用梯形法沿列方向求函数 y 关于自变量 x 的积分
Sct=cumtrapz(x, y)	采用梯形法沿列方向求函数 y 关于自变量 x 的累计积分

说明

- 假如 $\boldsymbol{X}$ 是 $(m\times n)$ 的数组，那么 sum(X)的计算结果 Sx 是一个 $(1\times n)$ 的数组，其中 $Sx(k)$ 就是 $\boldsymbol{X}$ 第 k 列全体元素的和。而 cumsum(X)的计算结果 Scs 仍是 $m\times n$ 数组，它第(i, k)元素，就是 $\boldsymbol{X}$ 数组第 k 列前 i 个元素的和。
- trapz(x,y)给出采样点(x, y)所连接折线下的面积，即函数 y 在自变量区间 x 上的近似积分。而 cumtrapz(x, y)的计算结果 Sct 是一个与 y 同样大小的数组，Sct(k)是 $\int_{x(1)}^{x(k)} y(x)\mathrm{d}x$ 的近似值。关于 cumtrapz 的算例还可见例 3.1-1 和例 3.1-2。
- 在对计算数值积分精度没有严格要求的场合，trapz 和 cumtrapz 是两条比较方便易用的指令。这两条指令所得数值积分的精度与积分区间分割的稀密程度有关。采样点数愈多，积分精度愈高，但精度无法定量控制。

例【4.1-4】 求积分 $s(x)=\int_0^{\pi/2} y(t)\mathrm{d}t$，其中 $y=0.2+\sin t$。本例演示：trapz 用于数值积分时的基本原理；sum 的用法及注意事项（见图 4.1-4）。

```
clear
d = pi/8;                          % 分区间的区间间隔
t = 0:d:pi/2;                      % 包含 5 个采样点的一维数组
y = 0.2 + sin(t);                  % 5 个点处的函数值数组
s = sum(y);                        % 求出的是：所有函数采样值之和
s sa = d * s;                      % 高度为函数采样值的所有小矩形面积之和          <6>
s ta = d * trapz(y);               % 连接各函数采样值的折线下的面积                <7>
disp(['sum 求得积分',blanks(3),'trapz 求得积分'])
disp([s sa, s ta])
t2 = [t,t(end) + d];                        % 因采用 stairs 绘图需要而写
y2 = [y,nan];                               % 因采用 stairs 绘图需要而写
stairs(t2,y2,':k')                          % 用虚线下面积表示 d * sum 的几何意义
hold on
plot(t,y,'r','LineWidth',3)                 % 用红折线下面积表示 d * trapz 的几何意义
h = stem(t,y,'LineWidth',2);                % 用蓝空心杆线表示函数采样值
set(h(1),'MarkerSize',10)
axis([0,pi/2 + d,0,1.5])                    % 使横坐标恰好是[0, 5 * d]
hold off
shg
sum 求得积分    trapz 求得积分
    1.5762      1.3013
```

图 4.1-4 sum 和 trapz 求积模式示意

说明

- 本例第〈7〉条指令可用更一般的格式 s_ta＝trapz(t,y)替换。这种通用格式可适用于"非等间隔采样"场合。该指令的几何意义是:计算由 t, y 所绘出折线下的面积。
- sum 指令是为求取数组元素和而设计的指令,也有人把它与自变量步长的乘积用作近似积分(如指令〈6〉)。但必须指出,这是一种误解。从图 4.1-4 可以清楚看到,阶梯虚线所占的自变量区间比积分区间多一个采样子区间。
- 为了揭示 trapz 实现梯形近似积分的几何意义,并揭示不能把 d * sum(y)看作"矩形近似积分"的理由,本例把子区间取得较大。在实际使用中,为了获得较高精度的近似积分,应该把子区间划分得相当小。

4.1.3 计算精度可控的数值积分

前面已经介绍了计算数值积分的近似指令。那些指令使用起来虽然简单,但既无法预先设置欲求积分的精度,也无法得知已求得积分的精度。本节将介绍的积分指令可以克服那些缺点。

数值积分有闭型(Closed-type)算法、开型(Open-type)算法之分。这两者的区别在于:是否需要计算积分区间端点处的函数值。MATLAB 提供的闭型数值积分指令如下:

S1＝quad(fun,a,b,tol)　　采用递推自适应 Simpson 法计算积分
S1＝quadl(fun,a,b,tol)　　采用递推自适应 Lobatto 法求数值积分
S2＝dblquad(fun,xmin,xmax,ymin,ymax,tol)　　二重(闭型)数值积分指令
S3＝triplequad(fun,xmin,xmax,ymin,ymax,zmin,zmax,tol)　　三重(闭型)数值积分指令

说明

- 被积函数**fun**,可以是字符串、内联对象、匿名函数和 M 函数文件的函数句柄。被积函数的自变量一般采用字母 x。在此提醒读者注意:编写被积函数时,要遵循"数组运算"规则。即保证:对于向量形式的自变量输入,输出的应是长度相同的函数值向量。
- 对于一重积分指令**quad, quadl**,其积分下限、上限由输入量**a**,**b** 传递。而多重积分指令**dblquad, triplequad** 的由内向外的积分限分别由输入量 **xmin, xmax, ymin, ymax, zmin, zmax** 传递。
- 输入宗量**tol** 是个标量,用来控制绝对误差。默认时,积分的绝对精度为 10^{-6}。

例【4.1-5】 求 $I=\int_0^1 e^{-x^2}dx$。

(1) 采用符号计算获得具有 32 位精度的积分值

```
syms x
Isym = vpa(int(exp( - x^2),x,0,1))
Isym =
.74682413281242702539946743613185
```

(2) 采用 trapz 计算积分

```
format long                              % 采用 15 位数字显示计算结果
d = 0.001;x = 0:d:1;
Itrapz = d * trapz(exp( - x. * x))
Itrapz =
```

```
   0.746824071499185
```

(3) 采用字符串表达被积函数

```
fx = 'exp( - x.^2)';                    % 字符串,关于串的说明请参见 A.1
Ic = quad(fx,0,1,1e - 8)                % 控制绝对精度达 1e - 8                <7>
Ic =
   0.746824132854452
```

说明

- 就本例而言,经与符号计算的 32 位精确积分值比较,知道采用 trapz 近似积分指令所得的结果 6 位数字有效。注意:事先并不能控制计算结果达到这样精度;计算后,假如不与更精确值比较,也是无法知道该近似积分的精度的。
- 采用 quad 等指令,可以事先指定数值积分的精度。比如第〈7〉条指令,指定积分值的绝对精度为 (1e-8)。观察结果可知,是积分值达到了 (1e-10)绝对精度。

例【4.1-6】 求 $s=\int_1^2\int_0^1 x^y\mathrm{d}x\mathrm{d}y$。本例演示:dblquad 指令的调用格式。

(1) 符号计算法

```
syms x y
s = vpa(int(int(x^y,x,0,1),y,1,2))                          %       <2>
Warning: Explicit integral could not be found.
s =
0.40546510810816438197801311546435
```

(2) 数值积分法

```
format long
s n = dblquad(@(x,y)x.^y,0,1,1,2)                           %       <4>
s_n =
   0.405466267243508
```

说明

- 当 dblquad 指令中的第 6 个输入量 tol 省略时,意味着保证数值积分的绝对精度为 (1e-6)。关于此,读者把输出结果与符号计算所得 32 位精度的结果进行比较就可看到。
- 本例 dblquad 指令第 1 个输入量采用"匿名函数"@(x,y)x.^y 写成。注意:在 dblquad 指令中,被积函数一定要写成"数组运算"格式。
- 对于本例而言,下列三条指令的作用相同。

```
s n = dblquad(@(x,y)x.^y,0,1,1,2)             % 采用匿名函数表示被积函数
s n = dblquad('x.^y',0,1,1,2)                 % 采用字符串表示被积函数
s n = dblquad(inline('x.^y'),0,1,1,2)         % 采用内联对象表示被积函数
```

4.1.4 函数极值的数值求解

对于函数极值问题,高等数学教科书给出的求解方法是:先对函数 $f(x)$求导函数 $f'(x)$,然后解方程 $f'(x)=0$ 得到满足方程的 x_0,进而通过分析 $f(x)$在 x_0 邻域内凹凸性,确定出 $f(x)$是否在 x_0 取得极值。这种极值确定法概念清晰、易于理解和接受。因此,这种极值确定

法不但在许多理论演绎中常被采用，而且在相当一些导函数比较简单的实际极值问题中也不少采用。MATLAB 的符号计算指令适用于这种场合。

这种极值确定法应用于实际时遇到两大障碍：一是，待求极值函数的导函数未必处处存在；二是，即使导函数存在，但 $f'(x)=0$ 的求解绝非轻而易举。

事实上，求极值的现代数值计算程序一般是利用由函数构成的某种“代价函数”值的不断下降原理进行搜索的。它们采用变步长、多项式（如抛物线）插补、黄金分割收缩区间等手段搜索函数的极值点。

本小节将介绍两条 MATLAB 求极小值的优化指令。顺便指出，MATLAB 只有处理极小值的指令，而没有专门针对极大值的指令。这是因为 $f(x)$ 的极大值问题等价于 $-f(x)$ 的极小值问题。

确切地说，这里只讨论“局域极值”问题。“全域最小”问题要复杂得多。至今没有一个“系统性”的方法可求解一般的“全域最小”问题。对于一元、二元函数，作图观察对确定全域最小有很好的应用价值。但更多元的函数，就很难使用作图法。下面是 MATLAB 求函数极值的两条指令。

[x,fval,exitflag,output]=fminbnd(fun,x1,x2,options)　　求一元函数在区间(x1, x2)中极小值

[x,fval,exitflag,output]=fminsearch(fun,x0,options)　　单纯形法求多元函数极值点

说明

- 第一输入量 fun 是待解目标函数，该目标函数可以采用字符串、内联对象、匿名函数和 M 函数文件的函数句柄等不同形式表达。**fminsearch** 被优化目标函数 fun 中的多元自变量应采用单一变量名的向量形式表达（见例 4.1-8）。
- **fminbnd** 的第二、三个输入量 x1、x2 分别表示被研究区间的左、右边界。输出量 x，fval 分别是极值点和相应的目标函数极值。
- **fminsearch** 的第二个输入量 x0 可以是一个搜索起点的向量或一组搜索起点的矩阵。当采用单个搜索起点时，输出量 x 也是一个单点（向量）。当采用多个搜索起点（矩阵）时，输出量 x 就给出多个搜索结果（矩阵），该矩阵的每一列代表一个候选极值点。这些搜索到的候选极值点按目标函数值递增次序排列。极值点 x(:,1) 对应的目标函数极小值由 fval 给出。
- 输入量 **options** 用于配置优化参数。它的默认值可用 **options＝optimset('FunFun_Name')** 察看。这里的 FunFun_Name 代表泛函指令名，例如 **fminbnd** 或 **fminsearch**。在没有特殊需求情况下，一般不必自行设置。
- 输出量 **exitflag** 若给出大于 0 的数，说明成功搜索到极值点。
- 输出量 **output** 给出具体的优化算法和迭代次数。
- 注意：从 MATLAB 7.0 版起，“泛函”指令不能通过输入量传递优化函数中的参数。
- 求一元函数极小值时，数值计算获得的区间收敛容差不会比 $\sqrt{\text{eps}}$ 更好。求多元函数极小值时，数值计算的精度极限是 $\frac{\|\Delta\vec{x}\|}{\|\vec{x}\|}<\sqrt{\text{eps}}$，$|\Delta f(\vec{x})|<\text{eps}$。

例【4.1-7】　已知 $y=e^{-0.1x}\sin^2 x-0.5(x+0.1)\sin x$，在 $-10\leqslant x\leqslant 10$ 区间，求函数的最小值。本例演示：符号计算求极值的局限性；fminbnd 求极小值的局限性；求最小值时，需要整个区间的函数信息和图形法功用。

(1) 用"导数为零"法求极值点

```
syms x
y = sin(x)^2 * exp( - 0.1 * x) - 0.5 * sin(x) * (x + 0.1);
yd = diff(y,x);                          % 求导函数
xs0 = solve(yd,x)                        % 求导函数为 0 的自变量值 xs0。        <4>
yd_xs0 = vpa(subs(yd,x,xs0),6)           % 验算用:导函数在 xs0 处为 0 吗?
y_xs0 = vpa(subs(y,x,xs0),6)             % 在 xs0 处的函数值
xs0 =
matrix([[0.050838341410271656880659496266968]])
yd_xs0 =
2.2958874039497802890014385492622 * 10^( - 41)
y_xs0 =
- 0.0012633177697419672454415434411 8
```

(2) 采用优化算法求极小值

```
x1 = - 10;x2 = 10;                       % 搜索区间的边界
yx = @(x)(sin(x)^2 * exp( - 0.1 * x) - 0.5 * sin(x) * (x + 0.1));
                                         % 采用匿名函数形式定义被求极小值的函数 y(x)
[xn0,fval,exitflag,output] = fminbnd(yx,x1,x2)                               %   <9>
                                         % xn0,fval 分别是极值点和函数极小值
xn0 =
   2.514797840754235
fval =
  - 0.499312445280039
exitflag =
     1
output =
    iterations: 13
     funcCount: 14
     algorithm: 'golden section search, parabolic interpolation'
       message: [1x112 char]
```

(3) 绘图观察最小值(参见图 4.1-5)

```
% 在[ - pi/2,pi/2]区间绘制函数 y(x)曲线
xx = - 10:pi/200:10;                     % 采样点应足够密
yxx = subs(y,x,xx);
plot(xx,yxx)
xlabel('x'),grid on
```

(4) 据图形观察,重设 fminbnd 的搜索区间

```
x11 = 6;x2 = 10;                         % 搜索区间的边界
yx = @(x)(sin(x)^2 * exp( - 0.1 * x) - 0.5 * sin(x) * (x + 0.1));
                                         % 采用匿名函数形式定义被求极小值的函数 y(x)
[xn00,fval,exitflag,output] = fminbnd(yx,x11,x2)                             %   <16>
                                         % xn0,fval 分别是极值点和函数极小值
```

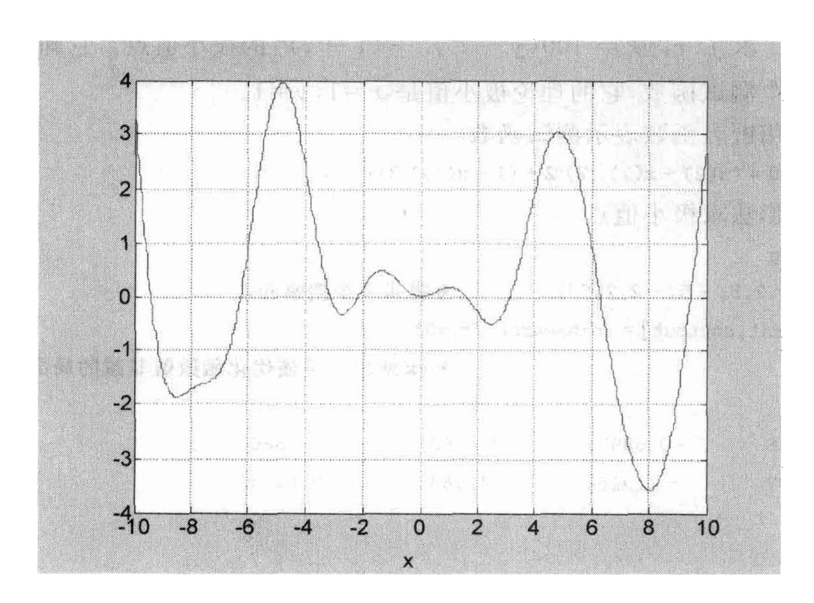


图 4.1-5　在[-10, 10]区间中的函数曲线

```
xn00 =
   8.023562824723015
fval =
  -3.568014059128578
exitflag =
     1
output =
    iterations: 9
     funcCount: 10
     algorithm: 'golden section search, parabolic interpolation'
       message: [1x112 char]
```

说明

- 在指定区间内，本例函数具有多个极小值。
- 指令〈4〉求出的"导函数为 0 方程"的解，即极小值点是 xs0，约等于 0.05；此时，相应的函数值 y_xs0 约等于 0。用户仅凭此解，无法断定它就是最小值点。
- 指令〈9〉借助 fminbnd 在[-10,10]之间搜索到得极小值点是 xn0，约为 2.5，对应的函数值 fval 约为 -3.56。显然，faval 比 y_xs0 小，因此 xn0 更可能是最小值点，但也无法作出断言。
- 借助绘制本例函数在[-10,10]内的图形，可以发现：最小值点在 x=8 附近。
- 根据图形观察，把搜索区间改为[6, 10]，再用 fminbnd 重新搜索，即指令〈16〉，可以得到指定区间 [-10, 10]内的真正最小值点 xn00，约等于 8.02。
- 通过本例，再次提醒读者：求最小值时，一定要注意"指定区间"；在可能的情况下，应尽量先借助图形观察函数，获取全局信息；不要错误地认为"符号计算给出的解就是所需的"；也不要错误地认为，fminbnd 一定有能力在指定的区间内找到真解。

例【4.1-8】 求 $f(x,y)=100(y-x^2)^2+(1-x)^2$ 的极小值点。它即是著名的 Rosenbrock's "Banana" 测试函数，它的理论极小值是 $x=1,y=1$。

（1）本例采用匿名函数表示测试函数

```
ff = @(x)(100 * (x(2) - x(1).^2)^2 + (1 - x(1))^2);
```

（2）用单纯形法求极小值点

```
format short g
x0 = [ - 5, - 2,2,5; - 5, - 2,2,5];                    % 提供 4 个搜索起点
[sx,sfval,sexit,soutput] = fminsearch(ff,x0)
                                                        % sx 给出一组使优化函数值非减的局部极小点
sx =
        0.99998       - 0.68971       0.41507        8.0886
        0.99997        - 1.9168        4.9643        7.8004
sfval =
   2.4112e - 010
sexit =
     1
soutput =
    iterations: 384
     funcCount: 615
     algorithm: 'Nelder - Mead simplex direct search'
       message: [1x196 char]
```

（3）检查目标函数值

```
format short e
disp([ff(sx(:,1)),ff(sx(:,2)),ff(sx(:,3)),ff(sx(:,4))])
  2.4112e - 010   5.7525e + 002   2.2967e + 003   3.3211e + 005
```

说明

- 注意：在编写目标函数时，自变量不是采用 x，y 表示，而是采用一个名为 x 的"二元向量"表示的。
- sx 和 sfval 分别给出一组极小值点坐标和它的目标值。这两个数组的第一列，分别给出（在已知搜索起点组的情况下的）最佳极小值点候选和相应的最小目标值。

4.1.5　常微分方程的数值解

MATLAB 为解决常微分方程初值问题提供一组设计精良、配套齐全、结构严整的指令，包括：微分方程解算（solver）指令、被解算指令调用的 ODE 文件格式指令、积分算法参数选项 options 处理指令以及输出处理指令等。本书为求简明，在此仅通过算例介绍最常用的 ode45 的基本使用方法。

[t,Y] = ode45(odefun,tspan,y0)　　　　采用 4 阶 Runge-Kutta 数值积分法解微分方程

说明

- 第一输入量 odefun 是待解微分方程的函数文件句柄。该函数文件的输出必须是待解函数的一阶导数。不管原问题是不是一阶微分方程组，当使用 ode45 求解时，必须转

化成（假设由 n 个方程组成）一阶微分方程组形式：

$$\boldsymbol{y}' = f(\boldsymbol{y}, t) \tag{4.1-1}$$

式中 $\boldsymbol{y}$ 是 $(n\times1)$ 向量。

- tspan 常被赋成二元向量 $[t_0, t_f]$，此时 tspan 用来定义求数值解的时间区间。
- 输入量 y0 是一阶微分方程组的 $n\times1$ 初值列向量。
- 输出量 t 是所求数值解的自变量数据列向量（假定其数据长度为 N），而 $\boldsymbol{Y}$ 则是 $(N\times n)$ 矩阵。输出量 $\boldsymbol{Y}$ 行中第 k 列 $\boldsymbol{Y}(:,k)$，就是式(4.1-1)中 $\boldsymbol{y}$ 第 k 分量的解。

例【4.1-9】 求微分方程 $\frac{d^2x}{dt^2}-\mu(1-x^2)\frac{dx}{dt}+x=0, \mu=2$，在初始条件 $x(0)=1, \frac{dx(0)}{dt}=0$ 情况下的解，并图示（见图 4.1-6 和图 4.1-7）。

(1) 把高阶微分方程改写成一阶微分方程组

令 $y_1=x, y_2=\frac{dx}{dt}$，于是原二阶方程可改写成如下一阶方程组

$$\begin{bmatrix} \frac{dy_1}{dt} \\ \frac{dy_2}{dt} \end{bmatrix} = \begin{bmatrix} y_2 \\ \mu(1-y_1^2)y_2 - y_1 \end{bmatrix}, \qquad \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

且设 $\mu=2$。

(2) 根据上述一阶微分方程组编写 M 函数文件 DyDt.m

```
function ydot = DyDt(t,y)
mu = 2;
ydot = [y(2);mu * (1 - y(1)^2) * y(2) - y(1)];
                              % 注意一阶导数 ydot 是(2 * 1)列向量
```

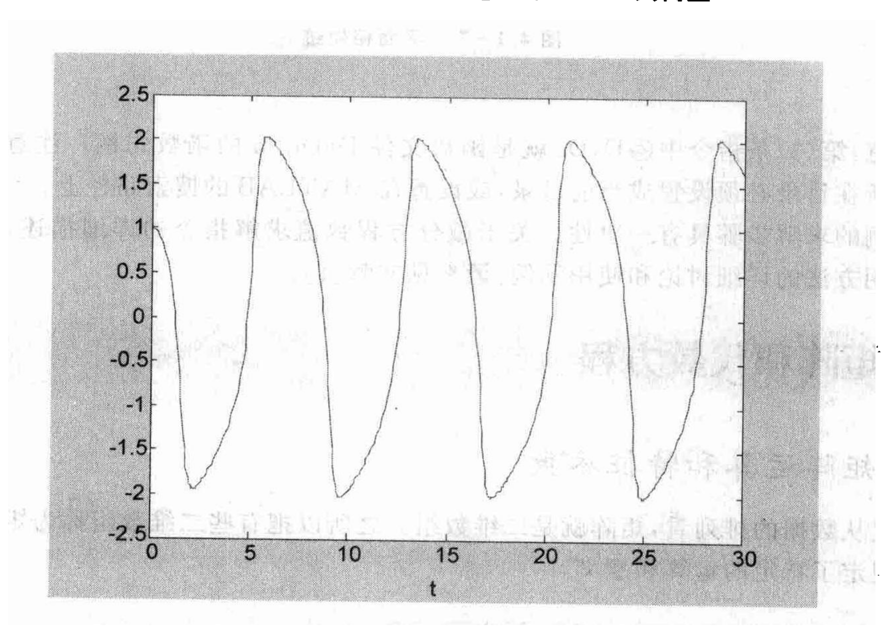


图 4.1-6　微分方程解

(3) 解算微分方程

```
tspan=[0,30];                  % 求解的时间区间
y0=[1;0];                      % 初值向量应与DyDt.m文件中y形式一致
[tt,yy]=ode45(@DyDt,tspan,y0);                                        % <3>
plot(tt,yy(:,1))
xlabel('t'),title('x(t)')
```

(4) 画相平面图(见图4.1-7)

```
plot(yy(:,1),yy(:,2))     % 函数和其导函数勾画的曲线称为"相轨迹"
xlabel('位移'),ylabel('速度')
```

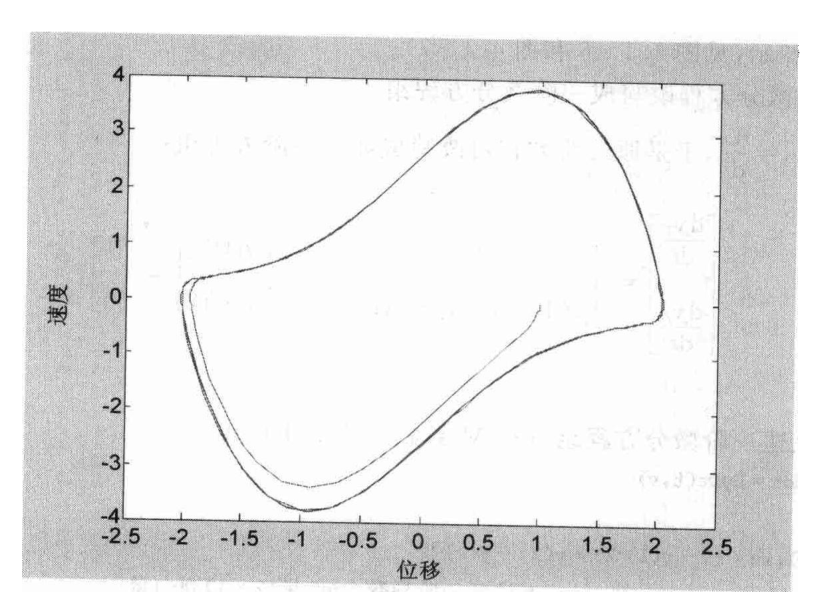


图 4.1-7 平面相轨迹

说明

- 注意:第〈3〉条指令中@DyDt就是函数文件DyDt.m的函数句柄。注意:DyDt.m文件所在目录必须设置成当前目录,或设置在MATLAB的搜索路径上。
- 本例的求解步骤具有一般性。关于微分方程数值求解指令的详细描述,ODE指令组使用方法的详细讨论和使用示例,请参见文献[1]。

4.2 矩阵和代数方程

4.2.1 矩阵运算和特征参数

如果仅从数据的排列看,矩阵就是二维数组。之所以把有些二维数组称为矩阵,是因为对这些数组规定了特定的运算和意义。

1. 矩阵运算

表 4.2-1　矩阵运算含义及相应符号

术　语	数学含义	MATLAB 表达
矩阵加减	$\boldsymbol{A}_{m\times n}\pm\boldsymbol{B}_{m\times n}=[a_{ij}\pm b_{ij}]_{m\times n}$	A+B,A−B
标量与矩阵加减	$a\pm\boldsymbol{B}_{m\times n}=[a]_{m\times n}\pm[b_{ij}]_{m\times n}=[a\pm b_{ij}]_{m\times n}$	a+B,a−B
矩阵乘积	$\boldsymbol{C}_{m\times n}=[c_{ij}]_{m\times n}=[a_{ij}]_{m\times l}\cdot[b_{ij}]_{l\times n}=\left[\sum_{k=1}^{l}a_{ik}b_{kj}\right]_{m\times n}=\boldsymbol{A}_{m\times l}\boldsymbol{B}_{l\times n}$	A * B
标量与矩阵相乘	$a\boldsymbol{B}_{m\times n}=[a\cdot b_{ij}]_{m\times n}$	a * B
矩阵的转置	$\boldsymbol{B}=\boldsymbol{A}^H\Leftrightarrow(b_{kj})_R+i\cdot(b_{kj})_I=(a_{jk})_R-i\cdot(a_{jk})_I$	B=A′

说明

- 在 MATLAB 中,“标量与矩阵加减运算”是一种“行业性”说法,在教科书中没有这种表述。本表格数学含义栏对“标量与矩阵加减运算”给出了完整的定义。
- MATLAB 的矩阵运算是在复数域定义的。在本表所列操作中,矩阵转置可全称为“矩阵的共轭转置(Hermition transpose)”,在数学表述上采用上标^H 表示。

例【4.2-1】 已知矩阵 $\boldsymbol{A}_{2\times4}$,$\boldsymbol{B}_{4\times3}$,采用三种不同的编程求这两个矩阵的乘积 $\boldsymbol{C}_{2\times3}=\boldsymbol{A}_{2\times4}\boldsymbol{B}_{4\times3}$。本例演示:矩阵乘法的含义;如何减少和避免循环;如何判断两个双精度矩阵是否相等。

(1) 实现矩阵乘的“标量程式”

矩阵乘积 $\boldsymbol{C}_{m\times n}=\boldsymbol{A}_{m\times l}\boldsymbol{B}_{l\times n}$的第$(i,j)$个元素的计算公式是

$$c_{ij}=\sum_{k=1}^{l}a_{ik}b_{kj}\qquad\forall i=1,\cdots,m;j=1,\cdots,n \tag{4.2-1}$$

该式以标量相乘为基础。下面就是据式(4.2-1),采用三重循环编写的计算乘积 $\boldsymbol{C}_{2\times3}$的程序。

```
clear
rand('twister',12)
A = rand(2,4);B = rand(4,3);
% ----------用三重循环体实现矩阵乘法----------
C1 = zeros(size(A,1),size(B,2));            % 为 C 预分配内存,可加快计算           <4>
for ii = 1:size(A,1)
    for jj = 1:size(B,2)
        for k = 1:size(A,2)
            C1(ii,jj) = C1(ii,jj) + A(ii,k) * B(k,jj);
        end
    end
end                                                                         % <13>
C1
C1 =
    0.7337    0.8396    0.3689
    1.0624    1.1734    1.3787
```

（2）实现矩阵乘的“saxpy 程式”

saxpy 是 LAPACK 计算术语，而 LAPACK 又是 MATLAB 内部函数的基本软件库。正是基于这种关系，介绍其计算机理即编程。乘法机理如下：

$$c_j = \sum_{k=1}^{l} a_k b_{kj} \qquad \forall j = 1,\cdots,n \tag{4.2-2}$$

该式以标量与向量相乘为基础，可减少循环次数。下面是采用二重循环编写的计算矩阵 $C_{2\times3}$ 的程序。

```
% ----------用二重循环体实现矩阵乘法----------
C2 = zeros(size(A,1),size(B,2));
for jj = 1:size(B,2)
    for k = 1:size(B,1)
        C2(:,jj) = C2(:,jj) + A(:,k) * B(k,jj);
    end
end
C2
C2 =
    0.7337    0.8396    0.3689
    1.0624    1.1734    1.3787
```

（3）MATLAB 中矩阵乘法的最好实现方式

MATLAB 以数组为计算单元，因此矩阵乘法可以直接进行，而无需进行任何循环。此外，这种程序与理论相乘表达式十分相似，便于阅读。具体指令如下。

```
C3 = A * B,                          % 直接运用 MATLAB 的矩阵乘
C3 =
    0.7337    0.8396    0.3689
    1.0624    1.1734    1.3787
```

（4）计算结果比较

三种编程方法呈现出一样的计算结果，这是观察判断的结论。考虑到数值计算途径不同会引起不同的截断误差，判断两个矩阵是否“数值意义上相等”的方法是：计算两个矩阵差的“范数”，若它等于零或接近 eps 数量级，则认为这两个矩阵“数值上相等”。下面就是判断指令。

```
C3_C3 = norm(C3 - C1,'fro')          % 采用计算矩阵 F 范数的方式比较两个矩阵之间的“差”
C3_C2 = norm(C3 - C2,'fro')          % 此值可能随计算机不同而变
C3_C3 =
     0
C3_C2 =
     0
```

说明

- “标量程式”的效率最差，saxpy 是 MATLAB 一系列提高计算效率措施中的一个，直接采用 MATLAB 中的矩阵乘算符“ * ”实现矩阵乘最好。
- 使用 MATLAB 的用户应该注意克服其他程序语言关于矩阵运算的编程习惯，因为 MATLAB 为矩阵运算设计的各种算符，不但简洁自然，而且计算效率和可靠性

都很好。

- 由于数值计算是有限精度计算，不同计算程式所得结果的比较一般采用矩阵范数进行。

例【4.2-2】 观察矩阵的转置操作和数组转置操作的差别。本例演示：共轭转置和非共轭转置；有理表示的显示格式。

```
format rat                         % 为简洁,采用有理格式显示
A = magic(2) + j * pascal(2)       % 仅为产生(2 * 2)复数矩阵
A =
      1     +   1i          3     +   1i
      4     +   1i          2     +   2i

% 注意两种不同的转置
A1 = A'                    % 共轭转置。这是一种矩阵操作
A2 = A.'                   % 非共轭转置。这是一种数组运算操作
A1 =
      1     -   1i          4     -   1i
      3     -   1i          2     -   2i
A2 =
      1     +   1i          4     +   1i
      3     +   1i          2     +   2i

% 注意操作符对结果的影响
B1 = A * A'
B2 = A. * A'
C1 = A * A.'
C2 = A. * A.'
B1 =
     12                    13     -   1i
     13     +   1i         25
B2 =
      2                    13     +   1i
     13     -   1i          8
C1 =
      8     +   8i          7     +  13i
      7     +  13i         15     +  16i
C2 =
      0     +   2i         11     +   7i
     11     +   7i          0     +   8i
```

2. 矩阵的标量特征参数

MATLAB 中用来计算(限于大学教材中涉及的)矩阵特征参数的指令，如表 4.2-2 所列。

表 4.2-2　计算矩阵标量特征参数的指令

术　语	数学含义	MATLAB 指令
秩 (Rank)	可采用以下任何一种表述： ● 矩阵 $\mathbf{A}$ 中线性无关列(或行)向量组中最大向量数； ● 矩阵 $\mathbf{A}$ 中最高非零子行列式的阶数； ● 矩阵 $\mathbf{A}$ 中最高非奇异子矩阵的维数	rank(A)
迹 (Trace)	$\sum_{i=1}^{\min(m,n)} a_{ii}$，即矩阵主对角元素之和	trace(A)
行列式 (Determinant)	$\lvert \mathbf{A}_{n\times n} \rvert = \sum_{j=1}^{n} (-1)^{j+1} a_{1j} \lvert \mathbf{A}_{1j} \rvert$， 式中 $\lvert \mathbf{A}_{1j} \rvert$ 是元素 a_{1j} 对应的子行列式	det(A)

例【4.2-3】 矩阵标量特征参数计算示例。本例演示：rank，det，trace 的使用；子行列式的计算。

```
A = reshape(1:9,3,3)          % 产生(3 * 3)矩阵
r = rank(A)                   % 求矩阵的秩
d3 = det(A)                   % 非满秩矩阵的行列式一定为 0
d2 = det(A(1:2,1:2))          % 求矩阵左上角(2 * 2)子行列式
t = trace(A)                  % 求矩阵的迹
A =
     1     4     7
     2     5     8
     3     6     9
r =
     2
d3 =
     0
d2 =
    -3
t =
    15
```

例【4.2-4】 矩阵标量特征参数的性质。本例演示：两相乘矩阵位置交换迹不变；两个同阶矩阵相乘位置交换行列式不变。

```
format short           % 采用 Notebook 默认显示格式
rand('twister',0)      % 设置均匀分布随机发生器的初始状态
A = rand(3,3);         % 生成(3 * 3)随机阵
B = rand(3,3);         % 生成另一个(3 * 3)随机阵
C = rand(3,4);
D = rand(4,3);

% 任何符合矩阵乘法规则的两个矩阵的乘积的"迹"不变
```

```
tAB = trace(A * B)        % 两个同阶矩阵乘积
tBA = trace(B * A)
tCD = trace(C * D)        % 两个"内维"相等矩阵的乘积
tDC = trace(D * C)
tAB =
        2.6030
tBA =
        2.6030
tCD =
        4.1191
tDC =
        4.1191

% 同阶矩阵乘积行列式等于各矩阵行列式之乘积
d_A_B = det(A) * det(B)
dAB = det(A * B)
dBA = det(B * A)
d_A_B =
        0.0094
dAB =
        0.0094
dBA =
        0.0094

% 对于非同阶矩阵,假如矩阵乘积存在,两个矩阵的乘积行列式随相乘次序不同而不同
% 当(m * n)矩阵与(n * m)矩阵之积,或是(m * m)阵,或是(n * n)阵
% 若 m>n,则积(m * m)阵一定行列式为零
dCD = det(C * D)
dDC = det(D * C)               % (m * n)
dCD =
    0.0424
dDC =
  -2.6800e-018
```

4.2.2 矩阵的变换和特征值分解

[R, ci] = rref(A)	借助初等变换把 **A** 变换成行阶梯矩阵 **R**
X = null(A)	**A** 矩阵零空间的全部正交基,满足 **AX**=**0**
Z = orth(A)	**A** 矩阵值空间的全部正交基,满足 span(**Z**)=span(**A**)
[V, D] = eig(A)	**A** 矩阵的特征值、特征向量分解,使 **AV**=**VD**

说明

- 指令 rref 的输出量 ci 是个行数组。它的元素表示了矩阵 **A** 中线性独立"列"的序号。因此,length(ci)就是矩阵 **A** 的秩;**A**(:,ci)的所有列向量构成矩阵 **A** 的值空间。但

length(ci)法计算矩阵秩远不如 rank(A)准确、可靠;A(:,ci)决定的值空间也不如 orth(A)可靠。

- 本指令主要用于教学目的,其结果形式比较接近高校教科书提供的求秩方法。
- 指令 eig 只适用于特征值各异的矩阵。假如存在相同特征值,则应尝试采用实现 Jordan 分解的指令 jordan。

例【4.2-5】 行阶梯阵简化指令 rref 计算结果的含义。本例演示:rref 对矩阵的分解;rref 输出量的含义。

(1) 对 4 阶魔方阵进行 rref 分解

```
A = magic(4)                %产生一个试验矩阵
[R,ci] = rref(A)            %行阶梯分解
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
R =
     1     0     0     1
     0     1     0     3
     0     0     1    -3
     0     0     0     0
ci =
     1     2     3
```

(2) rref 输出量 ci 的含义

ci 是行向量。其元素表明:矩阵 **A** 的第 1,2,3 列向量是"基"。而"基"的数目,即行向量 ci 的长度,正是矩阵 **A** 的秩。因此,矩阵的秩可以通过下述指令获得。

```
r_A = length(ci)
r_A =
     3
```

(3) rref 输出量 R 的含义

如果矩阵 **A** 的各列线性独立,那么输出阵 **R** 由"坐标(列)向量"构成。所谓"坐标(列)向量"是指除一个元素为 1 外,其余元素均为 0 的向量。

如果矩阵 **A** 的列线性相关,那么输出阵 **R** 还包含非"坐标向量"的列。这非"坐标向量"列的元素正是相应 **A** 阵中那列用基向量线性表出的系数。具体到本例,**R** 的第 4 列的元素就是 **A** 阵第 4 列由前 3 列线性表出的系数。检验指令如下。

```
aa = A(:,1:3) * R(1:3,4)        %基向量的线性组合
err = norm(A(:,4) - aa)         %若为 0 或接近 eps,说明 aa 就是 A(:,4)
aa =
    13
     8
    12
     1
```

```
err =
     0
```

例【4.2-6】 矩阵零空间及其含义。本例演示：null 指令的使用和输出结果的含义。

假设 $\boldsymbol{X}_{n\times l}$ 是矩阵 $\boldsymbol{A}_{m\times n}$ 的零空间，即 $\boldsymbol{A}_{m\times n}\boldsymbol{X}_{n\times l}=\boldsymbol{0}_{m\times l}$。而且，矩阵 $\boldsymbol{A}_{m\times n}$ 的秩 $\text{rank}(\boldsymbol{A}_{m\times n})=n-l$。下面以一个(5×3)的矩阵为例，演示零空间的获得和性质。

```
A = reshape(1:15,5,3);               % 产生(5 * 3)试验矩阵
X = null(A)                          % 零空间
S = A * X                            % 检验是否为"零"
n = size(A,2);                       % 矩阵 A 的列数
l = size(X,2);                       % 矩阵 X 的列数,即零空间的维数
n - l = = rank(A)                    % A 的列数减零空间维数应该等于 A 的秩
X =
    0.4082
   -0.8165
    0.4082
S =
  1.0e-014 *
   -0.2665
   -0.1776
   -0.0888
   -0.0888
   -0.0888
ans =
     1
```

例【4.2-7】 简单实阵的特征值分解。本例演示：eig，cdf2rdf 指令的使用；计算结果的验证。

(1) 特征值分解

```
A = [1, -3;2,2/3]
[V,D] = eig(A)
A =
    1.0000   -3.0000
    2.0000    0.6667
V =
   0.7746               0.7746
   0.0430 - 0.6310i     0.0430 + 0.6310i
D =
   0.8333 + 2.4438i          0
        0               0.8333 - 2.4438i
```

(2) 把复数特征值对角阵 $\boldsymbol{D}$ 转换成实数块对角阵

```
[VR,DR] = cdf2rdf(V,D)
VR =
    0.7746         0
```

```
    0.0430     -0.6310
DR =
    0.8333     2.4438
   -2.4438     0.8333
```

(3) 分解结果的验算

```
A1 = V * D/V                        % 由于计算误差,可能产生很小的虚部
A1_1 = real(A1)                     % 采用 real 去除虚部
A2 = VR * DR/VR
err1 = norm(A - A1,'fro')
err2 = norm(A - A2,'fro')
A1 =
   1.0000 - 0.0000i   -3.0000
   2.0000 + 0.0000i    0.6667
A1_1 =
    1.0000    -3.0000
    2.0000     0.6667
A2 =
    1.0000    -3.0000
    2.0000     0.6667
err1 =
  7.0290e-016
err2 =
  4.4409e-016
```

4.2.3 线性方程的解

1. 线性方程解的一般结论

对于含 n 个未知数的 m 个方程构成的方程组 $\boldsymbol{A}_{m\times n}\boldsymbol{x}=\boldsymbol{b}$,它的解有以下几种可能:

- 当向量 $\boldsymbol{b}$ 在矩阵 $\boldsymbol{A}$ 列向量所张空间中,有准确解。
  a) 若 $n=r$,则解唯一。(在此 r 是矩阵 $\boldsymbol{A}$ 的秩。)
  b) 若 $n>r$,则解不唯一。(在此 r 是矩阵 $\boldsymbol{A}$ 的秩。)
- 当向量 $\boldsymbol{b}$ 不在矩阵 $\boldsymbol{A}$ 列向量所张空间中,则无准确解,但存在最小二乘解。
  a) 当 $\boldsymbol{A}$ 列满秩时,存在唯一的最小二乘解。
  b) 当 $\boldsymbol{A}$ 列不满秩时,存在最小范最小二乘解和最少非零元素最小二乘解。

2. 除法运算解方程

当矩阵 $\boldsymbol{A}$ 非奇异时,线性代数教科书常介绍的线性方程 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解法有:Cramer 法、逆阵法($\boldsymbol{x}=\boldsymbol{A}^{-1}\boldsymbol{b}$)和 Gaussian 消元法。但是当在 MATLAB 环境中解方程时,我们极力推荐运用"除法"解线性方程。

MATLAB 对一般线性方程的求解进行了精心的设计,并采用简单直观的"除法"算符表达。具体如下:

$x=A\backslash b$　　运用左除解方程 $Ax=b$

说明

- 指令中的斜杠“\”是“左除”符号。由于方程 $Ax=b$ 中，A 在变量 x 的左边，所以指令中的 A 必须在“\”的左边，切不可放错位置。
- 假若方程是 $xC=d$ 形式，那么将使用“右除”，即指令为 $x=d/C$。

例【4.2-8】 求方程 $\begin{bmatrix}1 & 5 & 9\\2 & 6 & 10\\3 & 7 & 11\\4 & 8 & 12\end{bmatrix}\cdot x=\begin{bmatrix}13\\14\\15\\16\end{bmatrix}$ 的解。本例演示：如何确定解的性状（唯一与否，准确与否）；如何求特解和齐次解；如何检查解的正确性。

（1）创建待解方程的 A 和 b

```
A = reshape(1:12,4,3);          % 方程系数矩阵 A
b = (13:16)';                   % 方程右边的列向量 b
```

（2）检查 b 是否在 A 的值空间中（由此确定解的性状：不唯一、准确解）

```
ra = rank(A)                    % A 的秩
rab = rank([A,b])               % 若 rank([A,b]) = rank(A),则 b 在 A 的列空间中
ra =
     2
rab =
     2
```

（3）求特解和通解，并对由它们构成的全解进行验算

```
xs = A\b;                       % 求出特解
xg = null(A);                   % 求出齐次方程解
c = rand(1);                    % 随机数
ba = A * (xs + c * xg)          % 计算 A 与“一个随机的全解”的乘积 ba
norm(ba - b)                    % 检查 ba 与 b 的接近程度
Warning: Rank deficient, rank = 2,  tol =    1.8757e-014.
ba =
   13.0000
   14.0000
   15.0000
   16.0000
ans =
  1.3054e-014
```

3. 矩阵逆

如果 $(n\times n)$ 矩阵 A 和 B，满足 $AB=I_{n\times m}$，那么 B 称作 A 的逆，并采用符号 A^{-1} 记述之。MATLAB 提供一个求矩阵逆的指令如下：

A_1=inv(A)　　求非奇异方阵 A 的逆，使 $A*A_1=I$

说明

- 在 MATLAB 中，矩阵逆的用途极其有限。
- 当方程 $\boldsymbol{Ax}=\boldsymbol{b}$ 的系数矩阵 $\boldsymbol{A}$ 非奇异、且维数较低、条件较好时，尚可以用矩阵逆求解该方程。但这也主要出于教学目的。
- 先求逆 $\boldsymbol{A}^{-1}$，再用逆与向量 $\boldsymbol{b}$ 相乘求得解，不但费时，而且会引入额外的误差。以普通的标量方程 $7x=21$ 为例，$x=\frac{1}{7}\times 21=(0.142857\cdots)\times 21$ 的"逆阵法"求解过程就显得很"笨拙"。相比之下，$x=\frac{21}{7}=3$ 的"直接除法"精炼简捷得多。

例【4.2-9】 "逆阵"法和"左除"法解恰定方程的性能对比。

(1) 为对比这两种方法的性能，先用以下指令构造一个条件数很大的高阶恰定方程

```
randn('state',0);
A = gallery('randsvd',300,2e13,2);      % 产生条件数为 2e13 的 300 阶随机矩阵
x = ones(300,1);                        % 指定真解
b = A * x;                              % 为使 Ax = b 方程一致，用 A 和 x 生成 b 向量。
cond(A)                                 % 验算矩阵条件数
ans =
  1.9978e + 013
```

(2) "求逆"法解恰定方程的误差、残差、运算次数和所用时间

```
tic                                     % 启动计时器 Stopwatch Timer
xi = inv(A) * b;                        % xi 是用"逆阵"法解恰定方程所得的解
ti = toc                                % 关闭计时器，并显示解方程所用的时间
eri = norm(x - xi)                      % 解向量 xi 与真解向量 x 的范 - 2 误差
rei = norm(A * xi - b)/norm(b)          % 方程的范 - 2 相对残差
ti =
    0.0302
eri =
    0.1003
rei =
    0.0053
```

(3) "左除"法解恰定方程的误差、残差、运算次数和所用时间

```
tic;
xd = A\b;                 % "左除"求方程解
td = toc
erd = norm(x - xd)
red = norm(A * xd - b)/norm(b)
td =
    0.0138
erd =
    0.0939
red =
  8.4835e - 015
```

说明

- 计算结果表明：除法求解不但速度快，而且精度高得多。对本例而言，逆阵法所得解完全不可信。这是因为本例系数矩阵 **A** 的条件数很大。
- 一般而言，对于精度为 10^{-16} 的 MATLAB 双精度体系而言，假若矩阵 **A** 的条件数为 10^q，那么“逆阵法”所得方程解的精度不会高于 10^{q-16}，即有效数字不会超过 $(16-q)$ 位十进制。
- 这里显示的计算时间与具体机器有关、与相关指令是否第一次运行有关。

4.2.4　一般代数方程的解

对于任意函数 $f(x)=0$ 来说，它可能有零点，也可能没零点；可能只有一个零点，也可能有多个甚至无数个零点。因此，很难说出一个通用解法。一般说来，零点的数值计算过程是：先猜测一个初始零点或该零点所在的区间；然后通过一些计算，使猜测值不断精确化，或使猜测区间不断收缩；直到达到预先指定的精度，终止计算。

解题步骤大致如下：

(1) 利用 MATLAB 作图指令获取初步近似解

具体做法：先确定一个零点可能存在的自变量区间；然后利用 plot 指令画出 $f(x)$ 在该区间中的图形；观察 $f(x)$ 与横轴的交点坐标，或者更细致些，用 zoom 对交点处进行局部放大再读数。借助 ginput 指令获得更精确些的交点坐标值。

(2) 利用 MATLAB 的如下“泛函”指令求精确解

```
[x,favl]=fzero(fun,x0)          求一元函数零点指令的最简格式
[x,fval]=fsolve(fun,x0)         解非线性方程组的最简单格式
```

说明

- 被积函数**fun**，可以是字符串、内联对象、匿名函数和 M 函数文件的函数句柄。被解函数的自变量一般采用字母 x。
- 被处理函数**fun**，直接取“M 函数文件”文件名字符串的调用方法，仍可沿用，但建议少用或不用。
- 由于**fzero** 是根据函数是否穿越横轴来决定零点的，因此本程序无法确定函数曲线仅触及横轴而不穿越的那些零点。如 $|\sin(x)|$ 中的所有零点；$(x-1)^2\sin x$ 中 $x=1$ 的零点。
- 第二输入量**x0** 是表示零点初始猜测。**x0** 可以是标量或二元向量。当**x0** 取标量时，该指令将在它两侧寻找一个与之最靠近的零点；当 x0 取二元向量 $[a,b]$ 时，该指令将在区间 $[a,b]$ 内寻找一个零点。
- 指令中，输出量**x** 是所求零点的自变量值。
- 第二输出量**fval** 是函数值。

例【4.2-10】 求 $f(t)=\sin^2 t\cdot e^{-0.1t}-0.5|t|$ 的零点。本例演示：fzero 指令的用法；如何利用内联对象构造被解函数；作图法在数值求解中的作用；符号计算求非线性函数的零点。

(1) 采用符号计算求解

```
syms t
ft = sin(t)^2 * exp( - 0.1 * t) - 0.5 * abs(t);
S = solve(ft,t)                         % 求零点位置                    <3>
```

```
ftS = subs(ft,t,S)                        % 验算零点 S 处的函数值
S =
matrix([[0]])
ftS =
0
```

(2) 数值法求解

(A) 使用内联对象表示被处理函数。

```
y_C = inline('sin(t).^2.*exp(-0.1*t)-0.5*abs(t)','t');
    % 为利用此内联对象计算绘图所需的函数值，所以函数表达式采用数组运算
```

(B) 作图法观察函数零点分布(见图 4.2-1)。

```
t = -10:0.01:10;        % 对自变量采样，采样步长不宜太大
Y = y_C(t);             % 在采样点上计算函数值
clf,
plot(t,Y,'r');
hold on
plot(t,zeros(size(t)),'k');      % 画坐标横轴
xlabel('t');ylabel('y(t)')
hold off
```

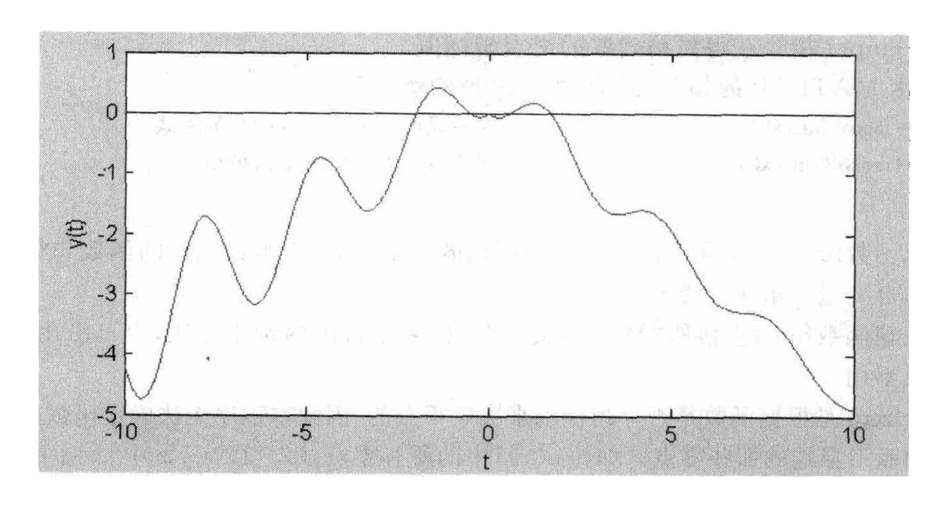


图 4.2-1 函数零点分布观察图

(C) 利用 zoom 和 ginput 指令获得零点的初始近似值。

由于 Notebook 中无法实现 zoom、ginput 指令涉及的图形和鼠标交互操作，因此下面指令必须在 MATLAB 指令窗中运行，并得到如图 4.2-2 所示的局部放大图及鼠标操作线。

```
zoom on                          % 在 MATLAB 指令窗中运行，获局部放大图
[tt,yy] = ginput(5);zoom off     % 在 MATLAB 指令窗中运行，用鼠标获 5 个零点猜测值
tt                % 显示所得零点初始猜测值(该指令可在 Notebook 中运行)
tt =
```

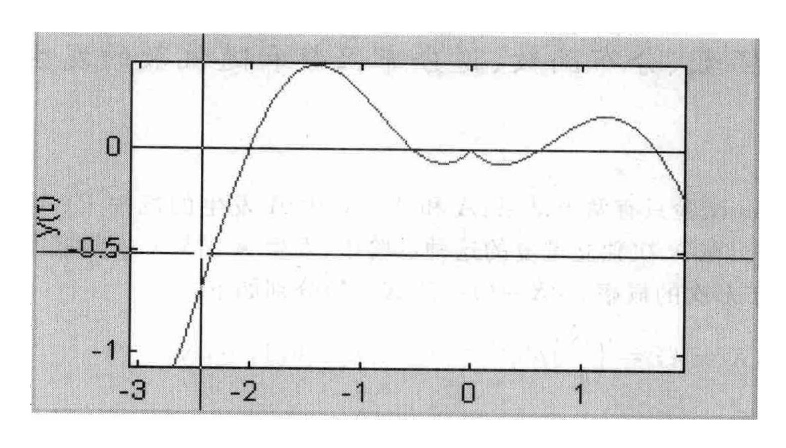


图 4.2-2　局部放大和利用鼠标取值图

```
  -2.0039
  -0.5184
  -0.0042
   0.6052
   1.6717
```

(D) 利用 0.1 作为初值求精确零点。

```
[t4,y4] = fzero(y_C,0.1)          % 以 0.1 位猜测值搜索附近的零点
t4 =
    0.5993
y4 =
  1.1102e-016
```

说明

- $f(t)$在$[-10, 10]$自变量区间有 5 个零点。指令〈3〉符号计算只能求得 $t=0$ 那个零点。
- 指令〈17〉采用 fzero 无法求得 $t=0$ 的那个零点。这是因为在 $t=0$ 处，$f(t)$没有穿越横轴。
- 图 4.2-2 中的十字是 ginput 运行后产生的取值图符。
- 关于高阶函数零点的求取，请见参考文献[1]。

4.3　概率分布和统计分析

概率统计在科学研究和工程应用中的地位日显重要，这在高校教学大纲中已有所体现。鉴于概率、统计、随机本身的特点，教材中一些在过去难以表达或体验的概念和算法，在计算机普及的今天就不再是障碍。

本节的内容包括：从离散和连续型随机变量概率分布中各挑选一种，介绍 MATLAB 中研究它们概率分布的指令；介绍一个便于读者增加对多种概率分布感性体验的交互界面；比较详细地介绍随机数的生成技术；介绍一组常用的统计指令。

4.3.1 概率函数、分布函数、逆分布函数和随机数的发生

1. 二项分布

每次 Bernoulli 试验只有两个结果:A 和 $\bar{A}$。其中 A 发生的概率 $P(A)=p$,$P(\bar{A})=q=1-p$,且 $0<p<1$。在 N 次独立重复的这种试验中,发生 A 结果 k 次的概率 $P(X=k)$ 和发生 A 结果次数不多于 k 次的概率 $F(X=k)=P(X\leqslant k)$ 分别如下:

$$P\{X=k\}=\binom{N}{k}p^k q^{(N-k)} \qquad k=0,1,\cdots,N$$

$$F(X=k)=P\{X\leqslant k\}=\sum_{j=0}^{k}\binom{N}{j}p^j q^{(N-j)} \qquad k=0,1,\cdots,N$$

服从以上函数关系的分布称为二项分布(binomial distribution),记为 $B(N,p)$。MATLAB关于二项分布的三个常用指令是:

pk=binopdf(k, N, p) 事件 A 发生 k 次的概率 $P\{X=k\}=\binom{N}{k}p^k q^{(N-k)}$

Fk=binocdf(k, N, p) 事件 A 发生次数不大于 k 的概率 $F(k)=\sum_{j=0}^{k}\binom{N}{j}p^j q^{(N-j)}$

R=binornd(N, p, m, n) 产生符合二项分布 $B(N,p)$ 的 $(m\times n)$ 随机数组 $\boldsymbol{R}$

说明

- 输入量 N 是独立重复试验的总次数;p 是在每次试验中结果 A 发生的概率;k 是在 N 次试验中结果 A 发生的次数,其取值必须在 $[0, N]$ 区间。
- 二项分布的 $E(k)=Np$,$D(k)=Npq$。
- 当二项分布的 $Np>5$,$N(1-p)>5$ 时,该分布就十分接近正态分布 $N(Np,(\sqrt{Npq})^2)$。

例【4.3-1】 画出 $N=100$,$p=0.5$ 情况下的二项分布概率特性曲线(见图 4.3-1)。本例演示:二项分布概率曲线形状;指令 binopdf 和 binocdf 的使用;双纵轴绘图指令 plotyy 指令的使用;低层图形指令对绘制图形的修饰。

```
N = 100;p = 0.5;                  % 给定二项分布的特征参数
k = 0:N;                          % 定义事件 A 发生的次数数组
pdf = binopdf(k,N,p);             % 算出各发生次数的概率
cdf = binocdf(k,N,p);             % 算出"不多于 k 次"事件的概率
h = plotyy(k,pdf,k,cdf);          % 采用双纵轴图纸画图
set(get(h(1),'Children'),'Color','b','Marker','.','MarkerSize',13)
                                  % 设置 pdf 曲线的颜色、数据点形状及大小
set(get(h(1),'Ylabel'),'String','pdf')
                                  % 书写左纵轴名称
set(h(2),'Ycolor',[1,0,0])
                                  % 设置右纵轴的颜色
set(get(h(2),'Children'),'Color','r','Marker','+','MarkerSize',4)
                                  % 设置 cdf 曲线的颜色、数据点形状及大小
```

```
set(get(h(2),'Ylabel'),'String','cdf')
                                        % 书写右纵轴名称
xlabel('k')                             % 横轴名称
grid on
```

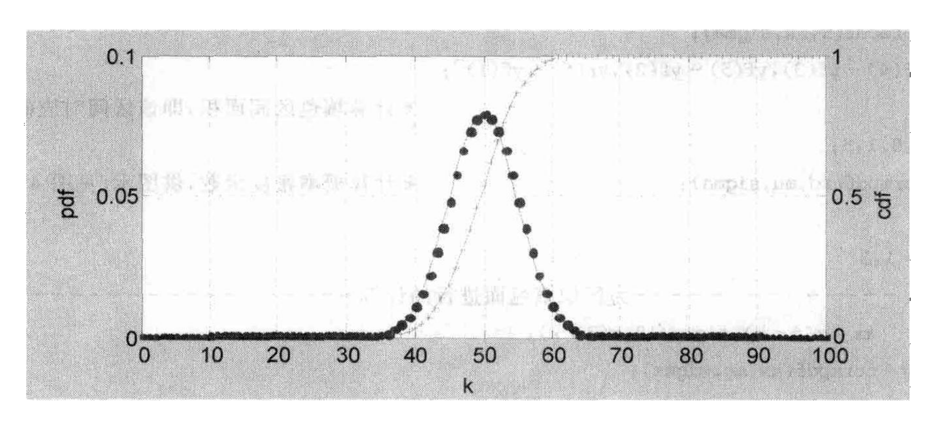


图 4.3-1　二项分布 B(100, 0.5)的概率和累计概率曲线

说明

- 从图 4.3-1 可见，若每次试验中，结果 A 发生的概率为 0.5，那么在所进行的 100 次独立重复试验中，获得 A 结果的试验次数最可能是 50，因为该二项分布的数学期望 $E(k)=100\times0.5$。但在 100 次试验中，A 结果恰出现 50 次的概率不到 0.08。
- 该二项分布很接近正态分布 $N(Np,(\sqrt{Npq})^2)$，具体为 $N(50,5^2)$。在 100 次试验中，A 结果发生的次数 k 最可能的取值范围是$[(50-3\times5),(50+3\times5)]$。图上曲线也说明了这种现象。

2. 正态分布

服从正态分布(normal distribution)$N(\mu,\sigma^2)$的连续型随机变量 X 的概率密度和累计概率密度函数分别为

$$f(x\mid\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}\qquad x\in(-\infty,+\infty)$$

$$F(x\mid\mu,\sigma)=\int_{-\infty}^{x}f(t\mid\mu,\sigma)\mathrm{d}t=\frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x}e^{\frac{-(t-\mu)^2}{2\sigma^2}}\mathrm{d}t$$

其中 μ,σ 分别是正态分布的数学期望和均方差(或称标准差)，即 $\mu=E(x),\sigma^2=D(x)$。

MATLAB 关于正态分布的三个常用指令是

px=normpdf(x, Mu, Sigma)	服从 $N(\mu,\sigma^2)$分布的随机变量取值 x 的概率密度
Fx=normcdf(x, Mu, Sigma)	服从 $N(\mu,\sigma^2)$分布的随机变量取值不大于 x 的概率
R=normrnd(Mu, Sigma, m, n)	产生元素服从 $N(\mu,\sigma^2)$分布的$(m\times n)$随机数组

说明

输入量 x 可以取任何实数；Mu 是正态分布的数学期望；Sigma 是正态分布的均方差。

【4.3-2】 正态分布标准差的几何表示(见图 4.3-2)。本例演示：normpdf, normcdf

的使用格式；指定区间的概率计算；标准差的含义和几何表示；绘图指令 fill 的应用；图形上希腊字母的书写。

```
mu = 3;sigma = 0.5;                                  % 设定均值和标准差
x = mu + sigma * [-3:-1,1:3];
yf = normcdf(x,mu,sigma);
P = [yf(4) - yf(3),yf(5) - yf(2),yf(6) - yf(1)];
                                                     % 计算填色区间面积，即该区间对应的概率
xd = 1:0.1:5;
yd = normpdf(xd,mu,sigma);                           % 计算概率密度函数，供图示(见图 4.3-2)
clf
for k = 1:3
    % ---------------为区域填色而进行的计算------------------
        xx = x(4 - k):sigma/10:x(3 + k);
    yy = normpdf(xx,mu,sigma);
    % ------------------------------------------------------
    subplot(3,1,k),plot(xd,yd,'b');                  % 画概率密度曲线
    hold on
    fill([x(4 - k),xx,x(3 + k)],[0,yy,0],'g');       % 给区间填色
    hold off
    if k<2
        text(3.8,0.6,'[{\mu} - {\sigma},{\mu} + {\sigma}]')
    else
        kk = int2str(k);
        text(3.8,0.6,['[{\mu} - ',kk,'{\sigma},{\mu} + ',kk,'{\sigma}]'])
    end
    text(2.8,0.3,num2str(P(k)));shg
end
xlabel('x');shg
```

说明

正态分布标准差的概率意义是：观察值 x 落在 $[\mu-\sigma,\mu+\sigma]$，$[\mu-2\sigma,\mu+2\sigma]$，$[\mu-3\sigma,\mu+3\sigma]$ 区间的概率，即 $P(\mu-k\cdot\sigma\leqslant x\leqslant\mu+k\cdot\sigma)$ 分别是 0.68269，0.9545，0.9973。由于 $P(\mu-k\cdot\sigma\leqslant x\leqslant\mu+k\cdot\sigma)=P(x-k\cdot\sigma\leqslant\mu\leqslant x+k\cdot\sigma)$，所以这概率意义又可以说成：测量数据两侧的一、二、三倍标准差区间包含该被测数据均值的概率分别是 0.68269，0.9545，0.9973。

3. 各种概率分布的交互式观察界面

作为优秀科学计算软件的 MATLAB 为从事科研工作的人员提供了各种直观易用的交互界面，概率分布交互界面就是其中之一。本节将以算例形式简单介绍该界面的使用。

例【4.3-3】 概率分布交互界面使用方法介绍。本例演示：disttool 指令引出的界面。

(1) 在指令窗中运行指令**disttool**，引出如图 4.3-3 所示界面。

(2) 在界面上方的 Distribution 选择栏中选中所需的分布名称(如正态分布 Normal 等)，

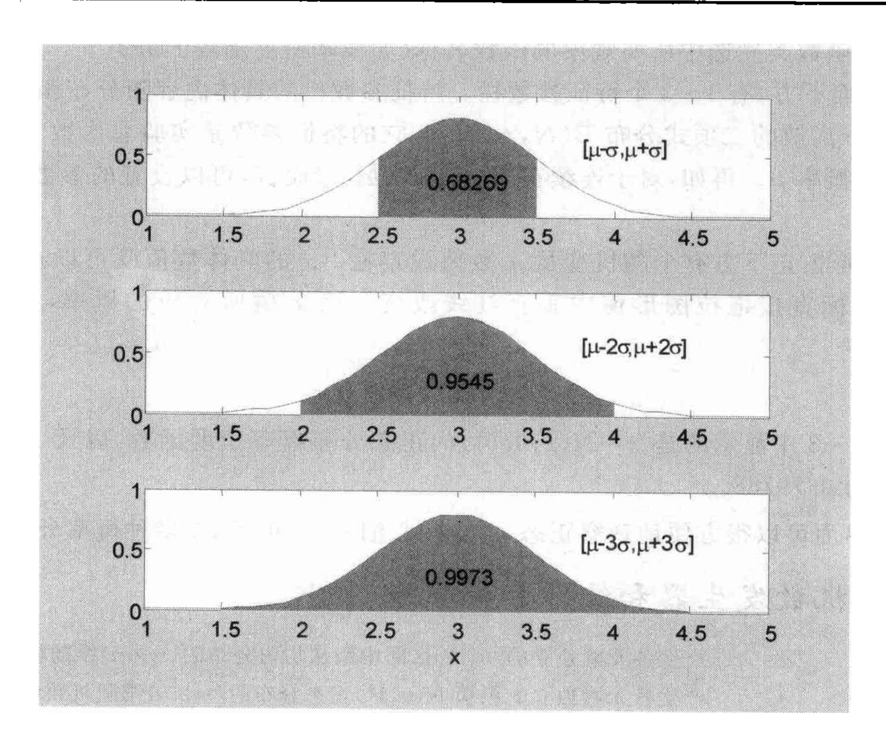


图 4.3-2　均值两侧一、二、三倍标准差之间的概率

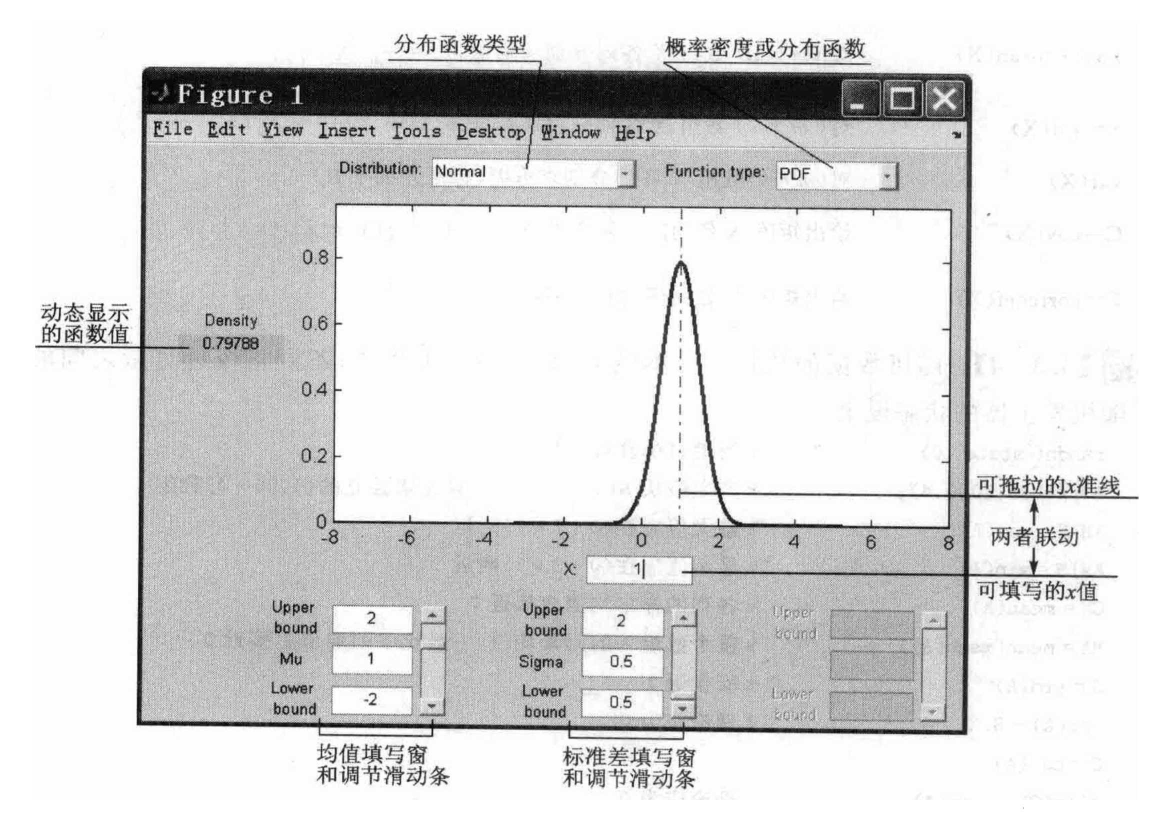


图 4.3-3　概率分布交互界面

再在右上方的函数类栏选中所要观察的函数名称(如概率密度函数 PDF)。

(3) 在界面下方,有 1～3 个特征参数栏。特征参数栏的具体内容随分布函数的不同而不同。例如,对于离散的二项式分布 $B(N,p)$ 而言,它的特征参数是实验总次数 N 和每次试验发生某结果的概率 p。再如,对于连续正态分布 $N(\mu,\sigma^2)$ 而言,可以设置的参数是数学期望 μ 和标准差 σ。

(4) 在图形窗正下方有个随机变量 x 数值设定栏。x 的具体数值既可以在该栏中输入,也可以通过鼠标直接拖拉图形窗中垂直红线改变。该 x 值所对应的概率,显示在图形窗的左侧。

说明

- 图 4.3-3 中显示的是 $x \sim N(1,0.5^2)$ 的正态分布概率密度函数,以及 $x=1$ 时的概率密度为 0.79788。
- 从该界面可以很方便地观察正态、χ^2、瑞利、泊松、二项等 20 多种概率分布。

4.3.2 随机数发生器和统计分析指令

rand(m,n)	产生各元素独立的[0, 1]区间中服从均匀分布的($m \times n$)维随机数组
randn(m,n)	产生各元素独立的服从 $N(0,1^2)$ 正态分布的($m \times n$)维随机数组
min(X)	对($m \times n$)数组 X 各列分别求最小值
max(X)	对($m \times n$)数组 X 各列分别求最大值
xbar=mean(X)	对($m \times n$)数组 X 各列分别求均值,$\bar{x}_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij}$
S=std(X)	对($m \times n$)数组 X 各列分别求标准差,$s_j = \left(\frac{1}{m-1}\sum_{i=1}^{m}(x_{ij}-\bar{x}_j)^2\right)^{\frac{1}{2}}$
var(X)	对($m \times n$)数组 X 各列分别求方差(标准差的平方)
C=cov(X)	给出矩阵 X 各列间的协方差阵。$c_{ij} = \frac{1}{m-1}(\boldsymbol{x}_i^{\mathrm{T}} - \bar{x}_i)(\boldsymbol{x}_j - \bar{x}_j)$
P=corrcoef(X)	给出矩阵 X 各列间的相关系数,即 $p_{ij} = \frac{c_{ij}}{\sqrt{c_{ii}c_{jj}}}$

例【4.3-4】 随机数据的统计量。本例演示:各指令的用法;不同指令计算量之间的关系;随机发生器的状态设定。

```
randn('state',0)                    % 为随机仿真可重复而设置
A = randn(1000,4);                  % 产生服从 N(0,1²)分布的各元素独立的(1000 * 4)数组
AMAX = max(A)                       % 最大值应在(0 + 3 * 1)附近
AMIN = min(A)                       % 最小值应在(0 - 3 * 1)附近
CM = mean(A)                        % 各列的样本均值应接近 0
MA = mean(mean(A))                  % 整个数组 A 的元素均值,一般比各列均值更接近 0
S = std(A)                          % 应接近 1
var(A) - S.^2                       % 理论应为 0
C = cov(A)
diag(C)' - var(A)                   % 理论应为 0
p = corrcoef(A)                     % 理论上应是单位阵
AMAX =
```

```
    2.7316      3.2025      3.4128      3.0868
AMIN =
   -2.6442     -3.0737     -3.5027     -3.0461
CM =
   -0.0431      0.0455      0.0177      0.0263
MA =
    0.0116
S =
    0.9435      1.0313      1.0248      0.9913
ans =
  1.0e-015 *
         0     -0.2220           0           0
C =
    0.8902     -0.0528      0.0462      0.0078
   -0.0528      1.0635      0.0025      0.0408
    0.0462      0.0025      1.0502     -0.0150
    0.0078      0.0408     -0.0150      0.9826
ans =
    0     0     0     0
p =
    1.0000     -0.0543      0.0478      0.0083
   -0.0543      1.0000      0.0024      0.0399
    0.0478      0.0024      1.0000     -0.0147
    0.0083      0.0399     -0.0147      1.0000
```

说明

- 当仿真中需要随机数样本可重复产生时，就必须对随机数发生器进行状态设置。状态设置指令为

  rand('twister',k)　　设置均布随机数发生器的初始状态为 k，(k 可取 0 和其他正整数)
  　　(注意：twister 是从 2008a 启用的，它产生的伪随机数的周期更长)

  randn('state',k)　　设置正态随机数发生器的初始状态为 k，(k 可取 0 和其他正整数)

- 当样本数有限时，样本统计量不会准确等于概率分布的相应数字特征，但一般比较接近。

例【4.3-5】 产生 1000 个服从 $N(2,0.5^2)$ 的随机数。本例演示：如何用 randn 产生指定均值和标准差的正态随机数；如何用 histfit 指令显示随机样本与理想正态分布的接近程度，参见图 4.3-4。

```
mu = 2;s = 0.5;
randn('state',22)
x = randn(1000,1);                                                    % <3>
y = s * x + mu;                   % 产生符合题意的随机样本               <4>
z = s * (x + mu);                 % 产生均值为 1 标准差为 0.5 的随机样本  <5>
subplot(3,1,1),histfit(x),axis([-5,5,0,100]),ylabel('x')
subplot(3,1,2),histfit(y),axis([-5,5,0,100]),ylabel('y')
subplot(3,1,3),histfit(z),axis([-5,5,0,100]),ylabel('z')
```

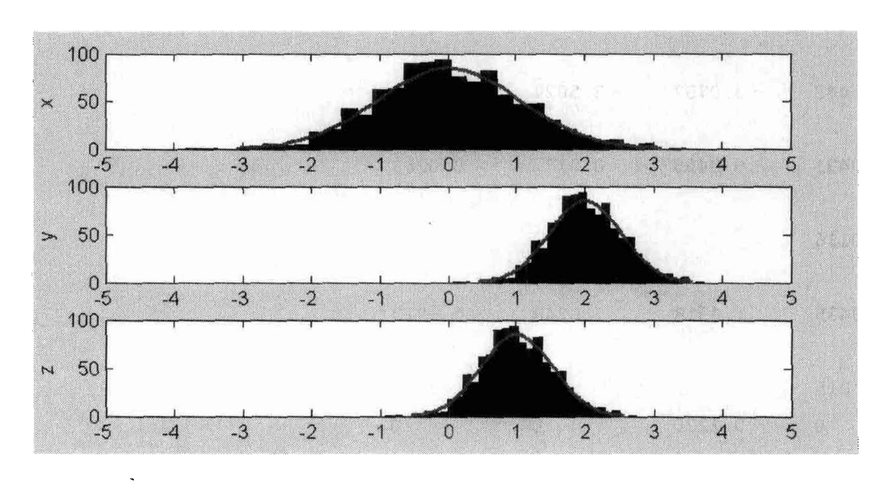


图 4.3-4 均值为 2 标准差为 0.5 的随机数样本 z

说明

- 服从 $N(2,0.5^2)$随机数产生的正确指令是第〈4〉条指令。本例之所以设计第〈5〉条指令是为了对比。
- hist 指令可以用来画随机数的频率直方图。在默认情况下，hist 把整个随机数取值区间分割成 10 个子区间；而 histfit 则把整个随机数取值区间分割成"样本数平方根"个子区间；每个子区间中所包含的随机数的个数，被显示为那子区间上直方条的高度。histfit 除画出直方图以外，还可自动计算所给数据的样本均值和样本标准差，然后画出相应的正态拟合曲线。
- 需要指出的是：运行指令 randn('state',22)；yy=normrnd(mu,s,1000,1)；可直接得到满足题目要求的随机数。

4.4 多项式运算和卷积

鉴于多项式计算和卷积在理工科教学、科研中的特殊地位和意义，专设一节予以阐述。

4.4.1 多项式的运算函数

1. 多项式表达方式的约定

MATLAB 约定降幂多项式 $a(x)=a_1x^n+a_2x^{n-1}+\cdots+a_nx+a_{n+1}$ 用以下系数行向量表示：$\boldsymbol{a}=[a_1,a_2,\cdots,a_n,a_{n+1}]$，即把多项式的各项系数依降幂次序排放在行向量的元素位置上。在此要提醒读者注意：假如多项式中缺某幂次项，则应认为该幂次项的系数为零。

2. 多项式运算函数

多项式运算函数的主要指令及其调用格式与含义如表 4.4-1 所列。

表 4.4-1　多项式运算函数指令的调用格式及含义

指　令	含　义
c=conv(a,b)	计算 $a(x)\times b(x)$ 乘积多项式 $c(x)$ 的系数向量 $\boldsymbol{c}$
[q,r]=deconv(b,a)	求出 $\frac{b(x)}{a(x)}=q(x)+r(x)\cdot\frac{1}{a(x)}$ 运算中商多项式 $q(x)$ 和余多项式 $r(x)$ 的系数向量 $\boldsymbol{q}$ 和 $\boldsymbol{r}$
[r,p,k]=residue(b,a)	当 $a(x)$ 不含重根时，计算部分分式分解 $\frac{b(s)}{a(s)}=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}+\cdots+\frac{r_n}{s-p_n}+k(s)$ 中的极点(Poles)、留数(Residues)和直项(Direct term)。 输出量 r 是由"各分子"构成的 $[r_1,r_2,\cdots,r_n]$ 一维数组； 输出量 p 是由"各极点"构成的 $[p_1,p_2,\cdots,p_n]$ 一维数组； 输出量 k 是多项式 $k(s)$ 的系数行向量
r=roots(a)	求 $a(x)$ 多项式的根
a=poly(r)	若 r 是一维数组，则该指令实施"据多项式根求多项式各项系数"的运算。此时，输入量 r 的各元素表示多项式的根，输出量 a 表示多项式的系数向量
	若 $\boldsymbol{r}$ 是方阵，则该指令实施"计算方阵特征多项式"的运算。此时，a 表示矩阵 r 所对应的特征多项式的系数向量
V=polyval(p,X)	在 $x=x_{ij}$ 时，计算多项式 $p(x)$ 的值 $v_{ij}=p(x_{ij})$。x_{ij} 是输入量 $\boldsymbol{X}$ 的第 (i,j) 元素；v_{ij} 是输出量 V 的第 (i,j) 元素。简单地说，按数组运算规则计算多项式值
V=polyvalm(p,X)	计算矩阵多项式的值 $\boldsymbol{V}=p_1\boldsymbol{X}^n+\cdots p_n\boldsymbol{X}+p_{n+1}\boldsymbol{I}$。换句话说，按矩阵运算规则计算多项式值。$p$ 为多项式，$\boldsymbol{X}$ 为矩阵

例【4.4-1】 求 $\frac{(s^2+2)(s+4)(s+1)}{s^3+s+1}$ 的"商"及"余"多项式。本例演示：多项式系数向量的正确表达；指令 conv，deconv，poly2str 的使用；如何验算。

(1) 求"商"及"余"多项式

```
format rat                          % 为避免浮点显示
p1 = conv([1,0,2],conv([1,4],[1,1]));      % 计算分子多项式             <2>
p2 = [1 0 1 1];                     % 定义分母多项式的系数向量。注意缺项补零
[q,r] = deconv(p1,p2);              % 求商多项式和余多项式
cq = '商多项式为   '; cr = '余多项式为   ';
disp([cq,poly2str(q,'s')]),disp([cr,poly2str(r,'s')])              <6>
商多项式为     s + 5
余多项式为     5 s^2 + 4 s + 3
```

(2) 利用计算所得"商"和"余"验算分子多项式

```
qp2 = conv(q,p2);                    % 计算"商"与"分母"的乘积
pp1 = qp2 + r; % 重算得到的分子多项式                                  <8>
pp1 = = p1                           % 对应系数相等，则结果为全 1        <9>
ans =

     1          1          1          1          1
```

说明

- 指令〈2〉使用了两次 conv。这里体现了乘法结合律。
- 指令〈6〉中的 poly2str 是一个函数文件，它在 MATLAB 的 Control Toolbox 中。它能据多项式系数向量写出比较易读形式的多项式。
- 本例中直接借助“==”比较 p1 和 pp1 是由于原多项式系数采用了整数，而且运算极其简单。一般说来，不同算法获得的浮点运算结果是不能借助“==”进行验算的，而必须通过两结果之“差”的“范数”大小加以判断。

例【4.4-2】 矩阵和特征多项式，特征值和多项式根。本例演示：poly，roots 的用法；矩阵特征值与特征多项式根的关系；多项式求根在 MATLAB 中是如何实现的；如何生成多项式的伴随矩阵。

（1）求矩阵的特征多项式

```
format short
A=[11 12 13;14 15 16;17 18 19];          % 创建一个试验用的(3*3)矩阵
PA=poly(A)                              % A 的特征多项式                    <2>
PPA=poly2str(PA,'s')                    % 以较习惯的方式显示多项式
PA =
    1.0000   -45.0000   -18.0000    0.0000
PPA =
   s^3 - 45 s^2 - 18 s + 1.6206e-014
```

（2）方阵特征值和特征多项式根

```
s=eig(A)
r=roots(PA)
s =
   45.3965
   -0.3965
    0.0000
r =
   45.3965
   -0.3965
    0.0000
```

（3）特征多项式的伴随矩阵

```
n = length(PA);                                  % 多项式系数向量的长度

AA  = diag(ones(1,n-2,class(PA)),-1);                            %     <7>
AA(1,:) = -PA(2:n) / PA(1);                      % 据多项式系数构成伴随阵    <8>
AA
sr  = eig(AA)
AA =
   45.0000   18.0000   -0.0000
    1.0000         0         0
         0    1.0000         0
```

```
sr =
    45.3965
    - 0.3965
     0.0000
```

说明

- n 阶方阵的特征多项式系数向量一定是 $1\times(n+1)$ 的，并且该系数向量第一个元素必是 1。这是因为指令 poly 输出的特征多项式经过“首项系数归一化”处理的缘故。
- 本例通过计算矩阵特征值 s 和特征多项式根 r，验证了“两者相同”的理论结论。
- 事实上，MATLAB 的多项式求根算法是借助所谓的伴随矩阵特征值计算实现的。这是因为该算法在所有多项式求根算法中最稳定、可靠。本例的第(3)部分指令就体现了这种算法思想。
- 指令〈7〉中的 ones(1,n－2,class(PA))产生$[1\times(n-2)]$的行数组，且该数组的数据类型与 PA 相同。然后，该数组[1, 1]在 diag 指令作用下，被设置在矩阵的“第 1 下次对角线”上，于是生成矩阵 $\begin{bmatrix}0 & 0 & 0\\1 & 0 & 0\\0 & 1 & 0\end{bmatrix}$。
- 指令〈8〉在指令〈7〉所生成矩阵基础上，再在该矩阵第 1 行写入多项式的“次最高项以下的所有系数”，构成所谓的伴随矩阵 **AA**。
- 在此，还可以顺便指出：矩阵的多项式是唯一的，但具有相同特征多项式的矩阵是无限的。就本例而言，可以肯定矩阵 **A** 和 **AA** 是相似的。

例【4.4－3】 构造指定特征根的多项式。

```
R = [ - 0.5, - 0.3 + 0.4 * i, - 0.3 - 0.4 * i];     % 创建根向量
P = poly(R)                                         % 构造与 R 对应的多项式
PR = real(P)                                        % 求 P 的实部
PPR = poly2str(PR,'x')
P =
     1.0000     1.1000     0.5500     0.1250
PR =
     1.0000     1.1000     0.5500     0.1250
PPR =
   x^3 + 1.1 x^2 + 0.55 x + 0.125
```

说明

- 要形成实系数多项式，则根向量中的复数根必须共轭成对。
- 含复数的根向量所生成的多项式系数向量(如 P)的系数有可能带截断误差数量级的虚部。此时可采用取实部的指令“real”把那很小的虚部滤掉。

例【4.4－4】 多项式求值指令 polyval 与 polyvalm 的本质差别。本例演示：polyval，polyvalm 的计算实质；验证“Caylay-Hamilton”定理。

(1) 给定多项式和(2×2)数组

```
clear
```

```
p=[1,2,3];                    % 多项式系数向量
poly2str(p,'x')
X=[1,2;3,4]                   % (2*2)数组
ans =
   x^2 + 2 x + 3
X =
      1        2
      3        4
```

(2) polyval 求值的本质

```
va=X.^2+2*X+3                 % 数组多项式求值                    <5>
Va=polyval(p,X)
va =
      6       11
     18       27
Va =
      6       11
     18       27
```

(3) polyvalm 求值的本质

```
vm=X^2+2*X+3*eye(2)           % 矩阵多项式求值                    <7>
Vm=polyvalm(p,X)
vm =
     12       14
     21       33
Vm =
     12       14
     21       33
```

(4) 验证"Caylay-Hamilton"定理

```
cp=poly(X);                   % 求矩阵 X 的特征多项式
poly2str(cp,'x')
cpXa=polyval(cp,X)            % 数组多项式的值
cpX=polyvalm(cp,X)            % 矩阵特征多项式的值
ans =
   x^2 - 5 x - 2
cpXa =
     -6       -8
     -8       -6
cpX =
   1.0e-015 *
    0.2220           0
         0      0.2220
```

说明

- 提请注意:第〈5〉、〈7〉两条指令在"平方项"和"常数项"上的差别。

- "Caylay-Hamilton"定理：任何一个矩阵满足它自己的特征多项式方程。这意味着 cpX 理论上应该为零。但由于浮点运算，截断误差使得 cpX 实际上是很小的元素构成的阵。

4.4.2 多项式拟合和最小二乘法

1. 多项式拟合

已知变量 x,y 之间的函数关系为

$$y = a_1x^n + a_2x^{n-1} + \cdots + a_nx + a_{n+1} \qquad (4.4-1)$$

现希望通过实验获得的一组 $\{x_i,y_i|i=1,2,\cdots,m\}$ 测量数据，确定出系数 $(a_1,a_2,\cdots,a_{n=1})$。这类问题就称为多项式拟合问题。MATLAB 求解该问题的指令是

p=polyfit(x,y,n)　　　　求 x，y 数组所给数据的 n 阶拟合多项式系数向量 p

说明

为保证较好的拟合效果，多项式阶数要取得适当。过低，可能残差较大；过高，拟合模型将包含噪声影响，通常要保证 $n<m$。观察数据图形曲线，有助于适当阶数的确定。

例【4.4-5】 给定数据组 x_0,y_0，求拟合三阶多项式，并图示拟合情况（参见图 4.4-1）。

```
%给定数据对
x0 = 0:0.1:1;
y0 = [-.447,1.978,3.11,5.25,5.02,4.66,4.01,4.58,3.45,5.35,9.22];
%求拟合多项式
n = 3;
P = polyfit(x0,y0,n)
%图示拟合情况
xx = 0:0.01:1;
yy = polyval(P,xx);
plot(xx,yy,'-b',x0,y0,'.r','MarkerSize',20)
```

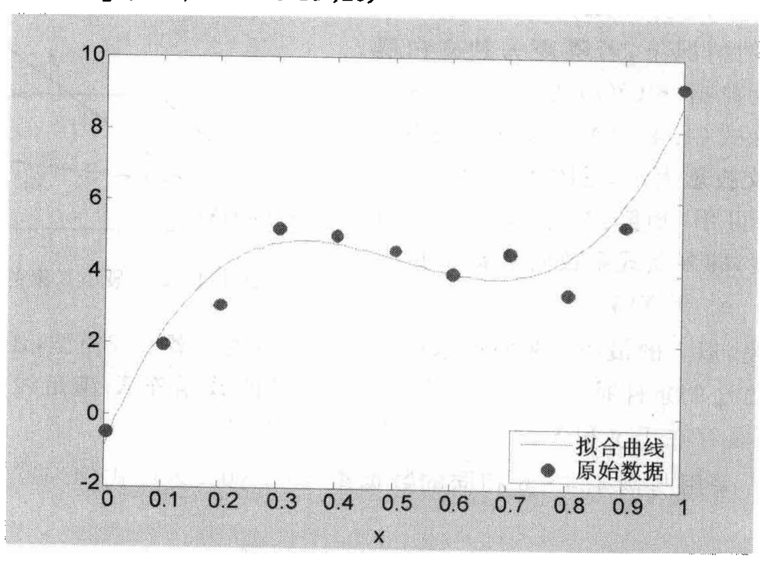


图 4.4-1　采用三阶多项式所得的拟合曲线

```
legend('拟合曲线','原始数据','Location','SouthEast')
xlabel('x')
P =
    56.6915   -87.1174    40.0070    -0.9043
```

说明

拟合多项式只能在给定数据所限定的区间内使用，不要任意向外拓展。

2. 最小二乘问题

多项式拟合问题可以更具体地表述如下：对于实验数据可写出一组方程：

$$
\begin{aligned}
y_1 &= a_1 x_1^n + a_2 x_1^{n-1} + \cdots + a_n x_1 + a_{n+1} + \varepsilon_1 \\
y_2 &= a_1 x_2^n + a_2 x_2^{n-1} + \cdots + a_n x_2 + a_{n+1} + \varepsilon_2 \\
&\vdots \\
y_m &= a_1 x_1^m + a_2 x_m^{n-1} + \cdots + a_n x_m + a_{n+1} + \varepsilon_m
\end{aligned}
$$

上述方程组可用矩阵形式简记为

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{a}^{\mathrm{T}} + \boldsymbol{\varepsilon} \tag{4.4-2}$$

在此 $\boldsymbol{y}=[y_1, y_2, \cdots, y_m]^{\mathrm{T}}$，$\boldsymbol{a}=[a_1, a_2, \cdots, a_{n+1}]$，$\boldsymbol{X}=\begin{bmatrix} x_1^n & \cdots & x_1 & 1 \\ x_2^n & \cdots & x_2 & 1 \\ \vdots & & \vdots & \vdots \\ x_m^n & \cdots & x_m & 1 \end{bmatrix}_{m\times n}$。式中的 $\boldsymbol{\varepsilon}$ 用来表述噪声（包括测量误差等）。多项式拟合问题就是由实验数据构成的 $\boldsymbol{y}$ 和 $\boldsymbol{X}$ 根据式（4.4-2）求多项式系数向量 $\boldsymbol{a}$。

在无噪声的情况下，由式（4.4-2）可知，$\boldsymbol{y}$ 是 $\boldsymbol{X}$ 的列向量线性组合。换句话说，$\boldsymbol{y}$ 在 $\boldsymbol{X}$ 的列所张的空间内，即 $\boldsymbol{y}\in \mathrm{span}\{\boldsymbol{X}\}$。

在存在噪声的情况下，若噪声为独立白噪声，且噪声与测量数据 x_i（$\forall i$）无关（它体现为 $E\{\boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{X}\}=0$），那么式（4.4-2）中的数学关系可形象地用几何正交投影表示，见图 4.4-2。

在 $m>n$ 时，可用“矩阵除”求取 $\boldsymbol{y}$ 在 $\mathrm{span}\{\boldsymbol{X}\}$ 上的投影长度，即多项式系数向量 $\boldsymbol{a}$ 如下：

$$\boldsymbol{a}^{\mathrm{T}} = \boldsymbol{X}\backslash\boldsymbol{y} \tag{4.4-3}$$

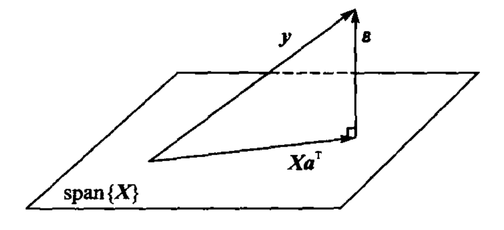


图 4.4-2 最小二乘的几何解释

需要指出的是：以上的最小二乘解的求取方法不仅适用于多项式模型，还适用其他更广泛的模型。使用该方法的条件是：只要应变量 y 与自变量的数据阵 X 满足线性关系就可。注意，这种线性关系是存在于 $\boldsymbol{y}$ 和 $\boldsymbol{X}$ 之间，而不是 y 与 x 之间。

例【4.4-6】 采用与例 4.4-5 相同的数据组 x0，y0，运用式（4.4-3）求拟合多项式的系数。

```
%给定数据组
x0 = (0:0.1:1)';
```

```
y0 = [ - .447,1.978,3.11,5.25,5.02,4.66,4.01,4.58,3.45,5.35,9.22]';
m = length(x0);
n = 3;                    % 多项式阶数
% 构成数据阵 X
X = zeros(m,n + 1);
for k = 1:n
    X(:,n - k + 1) = (x0.^k);
end
X(:,n + 1) = ones(m,1);
% 计算多项式系数
aT = (X\y0)'
aT =
   56.6915    - 87.1174    40.0070    - 0.9043
```

说明

显然，本例所得多项式系数与例 4.4－5 的结果相同。事实上，MATLAB 的 polyfit 文件就是采用本例思想编写的。

4.4.3　两个有限长序列的卷积

设有长度有限的两个任意序列

$$A(n)=\begin{cases}a_n & N_1\leqslant n\leqslant N_2\\ 0 & \text{else}\end{cases},\quad B(n)=\begin{cases}b_n & M_1\leqslant n\leqslant M_2\\ 0 & \text{else}\end{cases}$$

那么该卷积为

$$C(n)=\begin{cases}\sum_{i=N_1}^{N_2}A(i)B(n-i) & n\in[N_1+M_1,N_2+M_2]\\ 0 & \text{else}\end{cases}\qquad(4.4-4)$$

注意观察不难发现，卷积运算的数学结构与多项式乘法完全相同。正因为如此，MATLAB中的 conv，deconv 指令，不仅可用于多项式的乘除运算，而且可用于有限长序列的卷积和解卷运算。

例【4.4－7】 有序列 $A(n)=\begin{cases}1 & n=3,4,\cdots,12\\ 0 & \text{else}\end{cases}$ 和 $B(n)=\begin{cases}1 & n=2,3,\cdots,9\\ 0 & \text{else}\end{cases}$，求这两个序列的卷积。

（1）解法一："按卷积式(4.4－4)循环求和法"

```
N1 = 3;N2 = 12;
A = ones(1,(N2 - N1 + 1));          % 生成"非平凡区间"的序列 A
M1 = 2;M2 = 9;
B = ones(1,(M2 - M1 + 1));          % 生成"非平凡区间"的序列 B
Nc1 = N1 + M1;Nc2 = N2 + M2;        % 确定非平凡区间的自变量端点
kcc = Nc1:Nc2;                      % 生成非平凡区间的自变量序列 kcc
```

```
% 以下根据式(4.4-4)定义,通过循环求卷积
for n = Nc1:Nc2
    w = 0;
    for k = N1:N2
        kk = k - N1 + 1;
        t = n - k;
        if t> = M1&t< = M2
            tt = t - M1 + 1;
            w = w + A(kk) * B(tt);
        end
    end
    nn = n - Nc1 + 1;
    cc(nn) = w;                    % "非平凡区间"的卷积序列 cc
end
kcc,cc
kcc =
     5     6     7     8     9    10    11    12    13    14    15    16
17    18    19    20    21
cc =
     1     2     3     4     5     6     7     8     8     8     7     6
5     4     3     2     1
```

(2) 解法二:采用 conv 指令的"0 起点序列法"

```
N1 = 3;N2 = 12;
a = ones(1,N2 + 1);a(1:N1) = 0;      % 产生以 0 时刻为起点的 a 序列
M1 = 2;M2 = 9;
b = ones(1,M2 + 1);b(1:M1) = 0;      % 产生以 0 时刻为起点的 b 序列
c = conv(a,b);                       % 得到以 0 时刻为起点的卷积序列 c
kc = 0:(N2 + M2);                    % 生成从 0 时刻起的自变量序列 kc
kc,c
kc =
     0     1     2     3     4     5     6     7     8     9    10    11
12    13    14    15    16    17    18    19    20    21
c =
     0     0     0     0     0     1     2     3     4     5     6     7
8     8     8     7     6     5     4     3     2     1
```

(3) 解法三:采用 conv 指令的"非平凡区间序列法"

```
N1 = 3;N2 = 12;
M1 = 2;M2 = 9;
A = ones(1,(N2 - N1 + 1));          % 生成"非平凡区间"的序列 A
B = ones(1,(M2 - M1 + 1));          % 生成"非平凡区间"的序列 B
C = conv(A,B);                      % 得到"非平凡区间"的卷积序列 C
Nc1 = N1 + M1;Nc2 = N2 + M2;        % 确定非平凡区间的自变量端点
KC = Nc1:Nc2;                       % 生成非平凡区间的自变量序列 KC
```

```
KC,C
KC =
     5     6     7     8     9    10    11    12    13    14    15    16
17    18    19    20    21
C =
     1     2     3     4     5     6     7     8     8     8     7     6
5     4     3     2     1
```

(4) 绘图比较(见图 4.4-3)

```
subplot(2,1,1),stem(kc,c), text(20,6,'0 起点法')          % 画解法二的结果
CC=[zeros(1,KC(1)),C];                                     % 补零是为两子图一致
subplot(2,1,2),stem(kc,CC),text(18,6,'非平凡区间法')       % 画解法三的结果
xlabel('n')
```

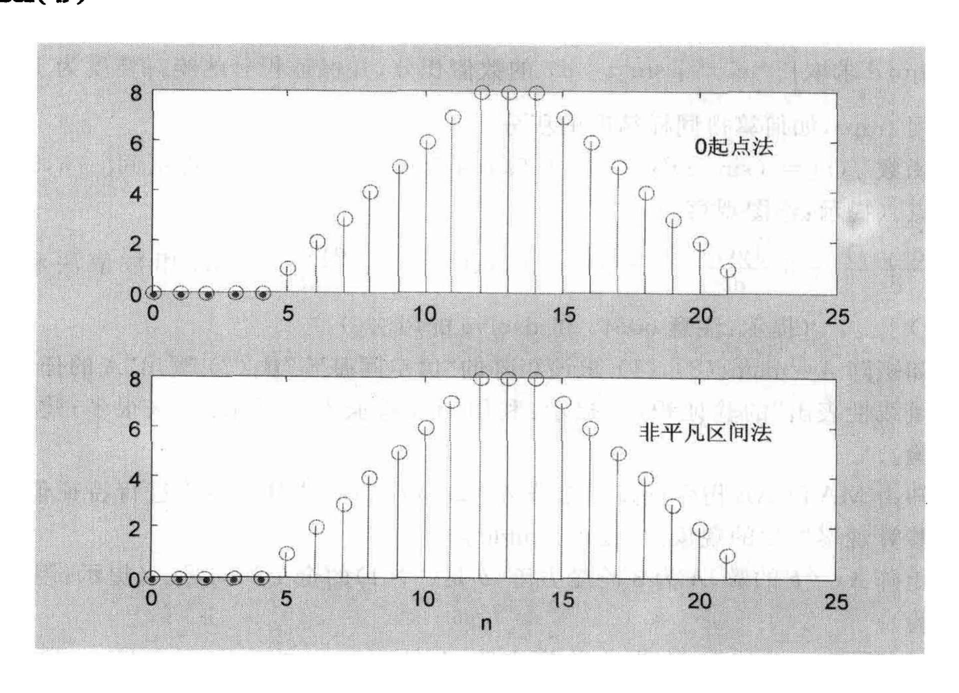


图 4.4-3 借助 conv 指令时两种不同序列记述法所得的卷积序列

说明

- 以上三种解法中优缺点:
  ① "解法三"最简洁、通用;
  ② "解法二"使用于序列起点时刻 N_1 或(和)M_1 小于 0 的情况,比较困难;
  ③ "解法一"最繁琐、效率低下。
- 关于无限长序列卷积、连续信号卷积和其他卷积方法的讨论,请见参考文献[1]。

习题 4

1. 根据题给的模拟实际测量数据的一组 t 和 $y(t)$ 试用数值差分 diff 或数值梯度 gradient 指令计算 $y'(t)$，然后把 $y(t)$ 和 $y'(t)$ 曲线绘制在同一张图上，观察数值求导的后果。(提示：模拟数据从 prob_data401.mat 获得。自变量 t 采样间距太小的影响。)
2. 采用数值计算方法，画出 $y(x)=\int_0^x \frac{\sin t}{t}\mathrm{d}t$ 在[0,10]区间曲线，并计算 $y(4.5)$。(提示：cumtrapz 快捷，在精度要求不高处可用；quad 也可试；要巧用 find。)
3. 求函数 $f(x)=\mathrm{e}^{\sin^3 x}$ 的数值积分 $s=\int_0^\pi f(x)\mathrm{d}x$，并请采用符号计算尝试复算。(提示：各种数值法均可试。)
4. 用 quad 求取 $\int_{-5\pi}^{1.7\pi}\mathrm{e}^{-|x|}\,|\sin x|\,\mathrm{d}x$ 的数值积分，并保证积分的绝对精度为 10^{-9}。(体验：试用 trapz，如何算得同样精度的积分。)
5. 求函数 $f(t)=(\sin 5t)^2\mathrm{e}^{0.06t^2}-1.5t\cos 2t+1.8\,|t+0.5|$ 在区间[−5,5]中的最小值点。(提示：作图观察。)
6. 设 $\frac{\mathrm{d}^2y(t)}{\mathrm{d}t^2}-3\frac{\mathrm{d}y(t)}{\mathrm{d}t}+2y(t)=1, y(0)=1, \frac{\mathrm{d}y(0)}{\mathrm{d}t}=0$，用数值法和符号法求 $y(t)\big|_{t=0.5}$。(提示：注意 ode45 和 dsolve 的用法。)
7. 已知矩阵 $\boldsymbol{A}$=magic(8)，(1) 求该矩阵的"值空间基阵"$\boldsymbol{B}$；(2) 写出"$\boldsymbol{A}$ 的任何列可用基向量线性表出"的验证程序(提示：利用 rref 检验)。(提示：方法很多；建议使用 rref 体验。)
8. 已知由 MATLAB 指令创建的矩阵 $\boldsymbol{A}$=gallery(5)，试对该矩阵进行特征值分解，并通过验算观察发生的现象。(提示：condeig)
9. 求矩阵 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解，$\boldsymbol{A}$ 为 3 阶魔方阵，$\boldsymbol{b}$ 是(3×1)的全 1 列向量。(提示：用 rref, inv, / 体验。)
10. 求矩阵 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解，$\boldsymbol{A}$ 为 4 阶魔方阵，$\boldsymbol{b}$ 是(4×1)的全 1 列向量。(提示：用 rref, inv, /体验。)
11. 求矩阵 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解，$\boldsymbol{A}$ 为 4 阶魔方阵，$\boldsymbol{b}=\begin{bmatrix}1\\2\\3\\4\end{bmatrix}$。(提示：用 rref, inv, /体验。)
12. 求 $-0.5+t-10\mathrm{e}^{-0.2t}|\sin[\sin t]|=0$ 的实数解。(提示：发挥作图法功用。)
13. 求二元函数方程组 $\begin{cases}\sin(x-y)=0\\ \cos(x+y)=0\end{cases}$ 的解。(提示：可尝试符号法解；试用 contour 作图求解；比较之。此题有无数解。)
14. 假定某窑工艺瓷器的烧制成品合格率为 0.157，现该窑烧制 100 件瓷器，请画出合格产品数的概率分布曲线。(提示：二项式分布概率指令 binopdf；stem。)
15. 试产生均值为 4，标准差为 2 的(10000×1)的正态分布随机数组 $\boldsymbol{a}$，分别用 hist 和

histfit 绘制该数组的频数直方图，观察两张图形的差异。除 histfit 上的拟合红线外，怎样使这两个指令绘出相同的频数直方图？（提示：体验 normrnd；理解 hist(Y, m) 指令格式。）

16. 从数据文件 prob_data416.mat 得到随机数组 R，下面有一段求取随机数组全部数据最大值、均值和标准差的程序。（提示：load；R(:)。）
Mx=max(max(R))，Me=mean(mean(R))，St=std(std(R))，试问该程序所得的结果都正确吗？假如不正确，请写出正确的程序。

17. 已知有理分式 $R(x)=\dfrac{N(x)}{D(x)}$，其中 $N(x)=(3x^3+x)(x^3+0.5)$，$D(x)=(x^2+2x-2)(5x^3+2x^2+1)$；(1) 求该分式的商多项式 $Q(x)$ 和余多项式 $r(x)$；(2) 用程序验算 $D(x)Q(x)+r(x)=N(x)$ 是否成立。（提示：采用范数指令 norm 验算。）

18. 现有一组实验数据 x，y（数据从 prob_data418.mat 获得），试求这组数据的 5 阶拟合多项式。（提示：load，polyfit，polyval。）

19. 已知系统冲激响应为 $h(n)=[0.05, 0.24, 0.40, 0.24, 0.15, -0.1, 0.1]$，系统输入 $u(n)$ 由指令 randn('state',1)；u=2*(randn(1,100)>0.5)-1 产生，该输入信号的起始作用时刻为 0。试用直杆图（提示：用 stem 指令）画出分别显示该系统输入、输出信号的两张子图，见图 P4-1。（提示：注意输入信号尾部的处理；NaN 的使用。）

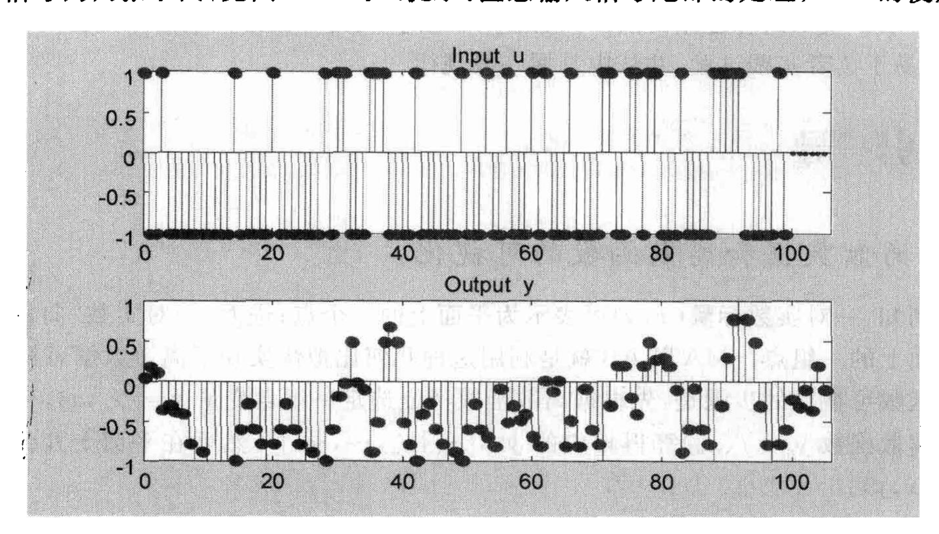


图 P4-1

第5章 数据和函数的可视化

视觉是人们感受世界、认识自然的最重要途径。数据可视化的目的在于:通过图形,从一堆杂乱的离散数据中观察数据间的内在关系,感受由图形所传递的内在本质。MATLAB 一向注重数据的图形表示,并不断地采用新技术改进和完备其可视化功能。

本章将系统地阐述:曲线、曲面绘制的基本技法和指令;如何使用线型、色彩、数据点标记凸现不同数据的特征;如何利用着色、灯光照明、烘托表现高维函数的性状;如何生成和运用标识,画龙点睛般地注释图形等。

本章的图形指令只涉及 MATLAB 的"高层"绘图指令。这种指令的形态和格式友善,易于理解和使用。整章内容遵循由浅入深、由基础到高级、由算例带归纳的原则。所有算例都是运行实例,易于读者实践试验,并从中掌握一般规律。

5.1 引 导

5.1.1 离散数据和离散函数的可视化

众所周知:一对实数标量(x,y)可表示为平面上的一个点;进而,一对实数"向量"$(\boldsymbol{x},\boldsymbol{y})$可表现为平面上的一组点。MATLAB 就是利用这种几何比拟法实现了离散数据可视化。

离散函数可视化的步骤是:先根据离散函数特征选定一组自变量$\boldsymbol{x}=[x_1,x_2,\cdots,x_N]^{\mathrm{T}}$;再根据所给离散函数$y_n=f(x_n)$算得相应的$\boldsymbol{y}=[y_1,y_2,\cdots,y_N]^{\mathrm{T}}$,然后在平面上几何地表现这组向量对$(\boldsymbol{x},\boldsymbol{y})$。

例【5.1-1】 用图形表示离散函数$y=|n|$。本例演示:自变量的适当选取;图形的适当比例;再次表现数组运算的简便有效;可视化只能表现有限区间(见图 5.1-1)。

```
n = ( - 10:10)';                  % 产生一组自变量数据
y = abs(n);                       % 函数的数组算法计算相应点的函数值
plot(n,y,'r.','MarkerSize',20)
axis equal
grid on                           % 画坐标格
xlabel('n')
```

说明

● 自变量取得关于 0 对称,是为了表现函数的对称性,表现函数变化趋势在 0 处发生突

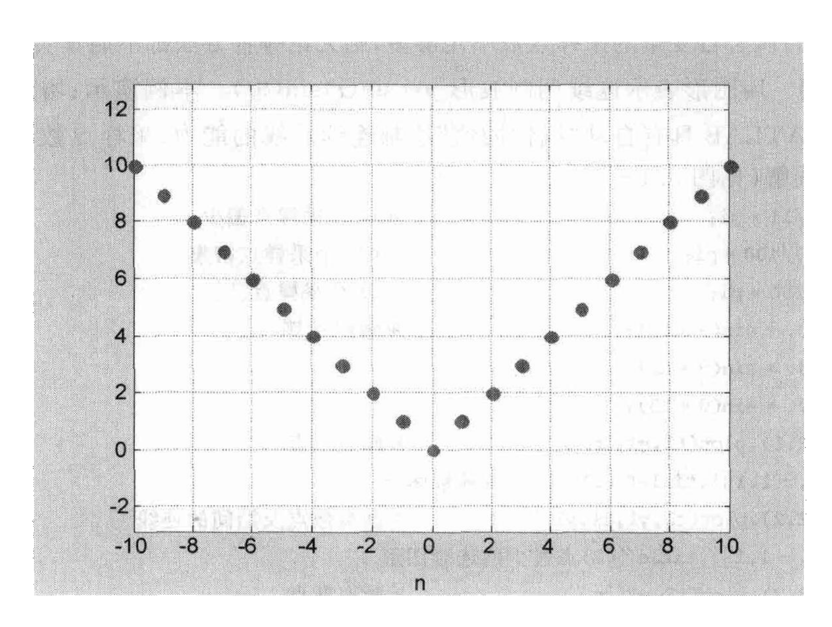


图 5.1-1　离散函数的可视化

然转折。

- 若自变量取为 $-20:0$，那么画出的图形显然没有很好反映 $y=|n|$ 的本质，因为 $y=-x$ 在这段自变量区间，也可画出相同的图形。
- 为显示离散点序列与横、纵坐标的夹角相等，采用了指令 axis equal。
- 函数完整地表现自变量与因变量之间的关系，可视化图形所表现的函数关系通常是局部的、非完整的。

5.1.2　连续函数的可视化

连续函数可视化包含三个重要环节：一是，从连续函数获得一组采样数据，即选定一组自变量采样点（包括采样的起点、终点和采样步长），并计算相应的函数值；二是，离散数据的可视化；三是，图形上离散点的连续化。

显然，图形上的离散点不能很好地表现函数连续性。进一步表示离散点之间的函数性状，有两种处理方法。

(1) 对区间进行更细的分割，计算更多的点，以近似表现函数的连续变化。这种方法的优点是：所画的每个点都反映真实的函数关系；缺点是：为了使图形上离散点密集到产生"连续感"，所需离散点的数量很大，从而大大增加计算负担。因此，在实际应用中，这种依靠增加离散点数量去获得"连续感"的方法较少采用。

(2) 在离散采样点的基础上，采用"线性插值"迅速算出离散点间连线上所经过的每个像素，从而获得"连续"曲线的效果。这种方法的优点是：曲线有良好的连续感，并且计算量小，绘图速度快；缺点是：除离散采样点外，所有连线都只是真实曲线的近似。此外，还需提醒的是：采用"插值连线"画图时，自变量采样点必须按单调增或单调减次序排列。

MATLAB 绘制连续曲线时，会根据用户指定的离散采样点，自动地进行插值计算，进而绘制出连续的曲线。

还值得指出：倘若自变量的采样点数不足够多，则无论哪种方法都不能真实地反映原函数。

例【5.1-2】 用图形表示连续调制波形 $y=\sin(t)\sin(9t)$。本例演示：增加图形"连续感"的两种方法；MATLAB 具有自动"线性插值"绘制连续曲线的能力；采样点数不够多会造成对所表现函数的误解(见图 5.1-2)。

```
t1 = (0:11)/11 * pi;                              % 12 个采样点偏少
t2 = (0:400)/400 * pi;                            % 401 个采样点密集
t3 = (0:50)/50 * pi;                              % 51 个采样点已够
y1 = sin(t1). * sin(9 * t1);                      % 数组运算
y2 = sin(t2). * sin(9 * t2);
y3 = sin(t3). * sin(9 * t3);
subplot(2,2,1),plot(t1,y1,'r.')                   % 画离散点                <7>
axis([0,pi, - 1,1]),title('(1)点过少的离散图形')
subplot(2,2,2),plot(t1,y1,t1,y1,'r.')             % 画离散点及期间的连线      <9>
axis([0,pi, - 1,1]),title('(2)点过少的连续图形')
subplot(2,2,3),plot(t2,y2,'r.')                   % 画离散点                <11>
axis([0,pi, - 1,1]),title('(3)点密集的离散图形')
subplot(2,2,4),plot(t3,y3)                        % 画连续曲线              <13>
axis([0,pi, - 1,1]),title('(4)点足够的连续图形')
```

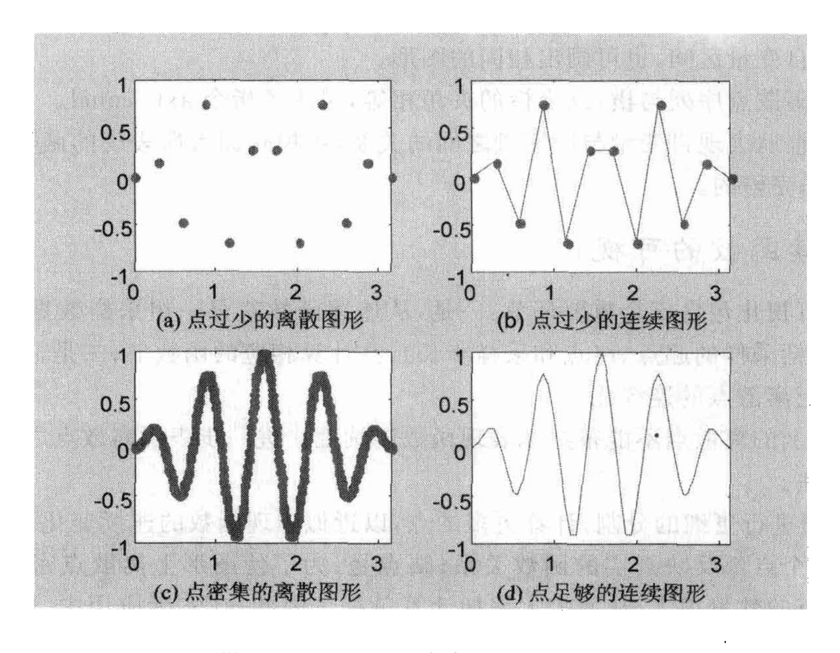


图 5.1-2 连续函数的图形表现方法

说明

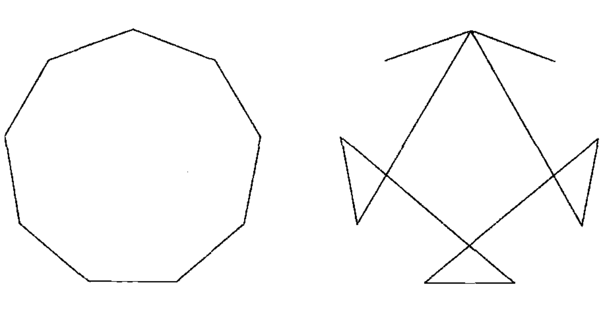
● 图 5.1-2 中子图(a)和(c)都是画离散点图形。显然，子图(a)由于 12 个采样点太少，看不出函数的性质。而子图(c)，虽然采样点有 401 个，但从图上看，仍然显得稀疏。

● 从子图(b)可以观察到两个事实:一,采样点只有 12 个,显然不足以反映函数的本来面貌;二,反映出 MATLAB 采用"线性插值"的实质,各采样点相连而成的"折线"。

● 子图(d),采样点数仅有 51 个,各采样点之间由直线连接。视觉上已感觉所画"折线"大致光滑地近似表现真实曲线(假如觉得所画折线不够光滑,还可适当增加采样点数)。

例【5.1-3】 绘制奇数正多边形及圆。本例演示:自变量单调排列对正确绘制连续曲线的重要性;如何画正多边形(见图 5.1-3)。

```
N=9;                                     % 多边形的边数
t=0:2*pi/N:2*pi;                         % 递增排列的自变量
x=sin(t);y=cos(t);                       % 参数方程,绘"奇数正多边形及圆"
tt=reshape(t,2,(N+1)/2);                 % 把行向量重排成"二维数组"
tt=flipud(tt);                           % 把"二维数组"的上下两行调换
tt=tt(:);                                % 获得变序排列的自变量
xx=sin(tt);yy=cos(tt);
subplot(1,2,1),plot(x,y)                 % 正常排序下的图形
title('(1) 正常排序图形'),axis equal off,shg
subplot(1,2,2),plot(xx,yy)               % 非正常排序下的图形
title('(2)非正常排序图形'),axis equal off,shg
```



(a) 正确排序图形　　　　(b) 非正常排序图形

图 5.1-3 自变量排列次序对连续曲线图形的影响

说明

绘制连续曲线时,自变量必须按递增或递减的次序排列,否则所画的曲线将发生异常。

5.2 二维曲线和图形

MATLAB 提供了多种二维图形的绘制指令(见表 5.2-1),但其中最重要、最基本的指令是 plot。其他许多特殊绘图指令,或以它为基础而形成,或使用场合较少。出于简明考虑,本节着重介绍 plot 的使用。

表 5.2-1 MATLAB 提供的二维图形绘制指令

指令名	含义和功能	指令名	含义和功能
area	面域图;主用于表现比例、成份	plot	基本二维曲线图形指令
bar	直方图;主用于统计数据	polar	以极坐标绘制曲线
compass	射线图;主用于方向和速度	quiver	二维箭头图;主用于场强、流向
feather	羽毛图;主用于速度	rose	频数扇形图;主用于统计
hist	频数直方图;主用于统计	stairs	阶梯图;主用于采样数据
pie	二维饼图;统计数据极坐标形式	stem	二维杆图;主用于离散数据

5.2.1 二维曲线绘制的基本指令 plot

1. 基本调用格式 plot(x,y,'s')

这是 plot 指令的最典型、最基本的调用格式。该指令的输入量(x ,y ,'s')称为平面绘线三元组。它们分别指定平面曲线的几何位置、点形、线型和色彩。

- **x**, **y** 是长度相同的一维数组。**x**, **y** 分别指定采样点的横坐标和纵坐标。
- 第三个输入量's'是字符串。它用来指定“离散点形”或/和“连续线型”(见表 5.2-2 及表 5.2-3),与此同时还可指定“点线色彩”(见表 5.2-4)。
- 假如 plot 指令没有第三个输入量,即's'不加指定,那么 plot 将使用默认设置——“蓝色细实线”绘制曲线。

表 5.2-2 离散数据点形允许设置值

符 号	含 义	符 号	含 义	符 号	含 义
d	菱形符 diamond	x	叉字符	<	朝左三角符
h	六角星符 hexagram	.	实心黑点	>	朝右三角符
o	空心圆圈	+	十字符	v	朝下三角符
p	五角星符 pentagram	*	米字符		
s	方块符 square	^	朝上三角符		

表 5.2-3 连续线型允许设置值

符 号	含 义	符 号	含 义
-	细实线	-.	点划线
:	虚点线	--	虚划线

表 5.2-4 点线色彩允许设置值

符 号	b	g	r	c	m	y	k	w
含 义	蓝	绿	红	青	品红	黄	黑	白

2. plot 的衍生调用格式

(1) 单色或多色绘制多条曲线

plot(X,Y,'s')	用 s 指定的点形线型色彩绘制多条曲线
plot(X,Y)	采用默认的色彩次序用细实线绘制多条曲线

说明

- 当 X, Y 均为($m\times n$)数组时,将绘制出 n 条曲线。每条曲线的几何位置由 X, Y 对应的列确定。
- 当 X, Y 两个输入量中有一个是一维数组,且该数组的长度与另一个输入量的"行数"(或"列数")相等时,将绘制出"列数"(或"行数")条曲线。
- plot(X,Y,'s')只能用 s 指定的同一色彩绘制多条曲线。
- 当 X, Y 为长度相同的一维数组时,该指令就是基本调用指令 plot(x,y,'s')。
- plot(X,Y)指令绘线采用细实线,并按照表 5.2-4 所列次序(除白色外)设置线的颜色,以提高"可观察性"。

(2) 多三元组绘制多条曲线

plot(X1,Y1,'s1',X2,Y2,'s2', ... ,Xn,Yn,'sn')

说明

- 该指令输入量由多个"三元组"(Xn ,Yn ,'sn')组成。
- 每个三元组是独立的,它的工作方式与 plot(X,Y,'s')完全相同。

(3) 单输入量绘线

plot(Y)

说明

- 当 Y 是一维数组时,以该数组的下标为横坐标、Y 为纵坐标绘制一条曲线。
- 当 Y 二维数组时,以该数组的"行下标"为横坐标、Y 为纵坐标绘制"列数"条曲线。
- 在 MATLAB 工作界面上,通过菜单选择可视化数组数据的曲线就是采用 plot(Y)这种格式绘制的。

3. plot 的属性可控调用格式

plot 绘制的曲线是图形对象之一,称为"线"。该线的几何形状及色彩都是它的属性表现。用户可以通过以下调用格式对绘线的属性加以控制。

plot(x, y, 's', 'PropertyName', PropertyValue, …)

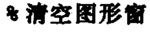
说明

- 运用属性名(PropertyName)和属性值(PropertyValue)可以对"线"对象的属性进行设置,使所绘曲线更具个性化。"线"图形对象最常用的属性见表 5.2-5。
- MATLAB 出于使用频率和方便考虑,采用's'字符串方式设置点形线型及几种常用色彩。当然,假若用户不采用's'字符串设置,而直接采用属性名/属性值对进行设置也是可以的,并且可以设置得更细腻。

表 5.2-5　线对象的常用属性名和属性值

含　义	属性名	属性值	说　明
点、线色彩	Color	$[v_r, v_g, v_b]$，RGB三元组中每个元素可在[0,1]取任意值。	● 最常用的色彩可通过表 5.2-3 中的字母表示 ● 常用色彩可通过's'设置。蓝色为默认色彩
线　型	LineStyle	4种线型参见表 5.2-3	● 可通过's'设置。细实线为默认线型
线　宽	LineWidth	正实数	● 默认线宽为 0.5
数据点形	Marker	14种点形参见表 5.2-2	● 可通过's'设置
点的大小	MarkerSize	正实数	● 默认大小为 6.0
点边界色彩	MarkerEdgeColor	$[v_r, v_g, v_b]$，RGB三元组中每个元素可在[0,1]取任意值	
点域色彩	MarkerFaceColor	$[v_r, v_g, v_b]$，RGB三元组中每个元素可在[0,1]取任意值	

例【5.2-1】 二维曲线绘图指令演示之一。本例演示：plot(t,Y) plot(Y)所绘曲线的区别；“线宽”属性的设置（见图 5.2-1）。



```
clf                                     % 清空图形窗
t=(0:pi/50:2*pi)';                      % 生成(101×1)的列向量
k=0.4:0.1:1;                            % 生成(1×7)的行向量
Y=cos(t)*k;                             % 生成(101×7)的矩阵
subplot(1,2,1),plot(t,Y,'LineWidth',1.5)
title('By plot(t,Y)'),xlabel('t')
subplot(1,2,2),plot(Y,'LineWidth',1.5)
title('By plot(Y)'),xlabel('row subscript of Y')
```

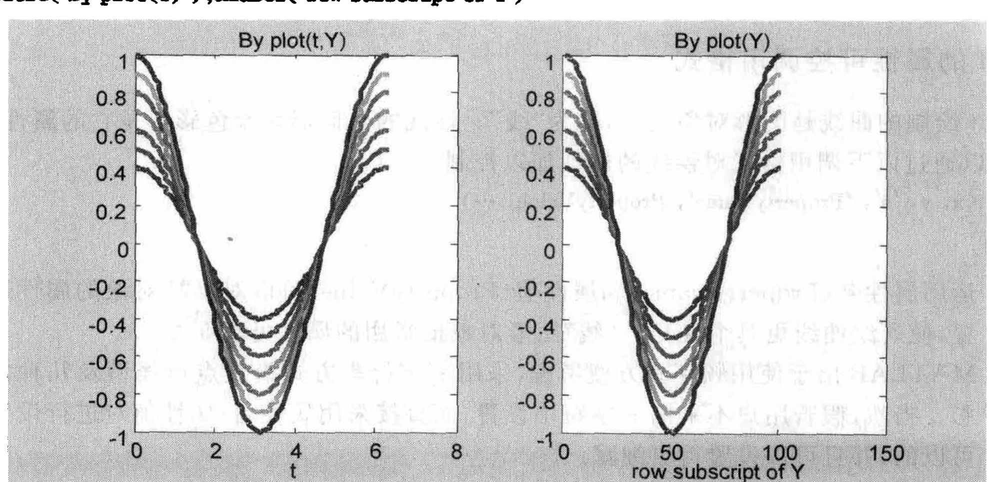
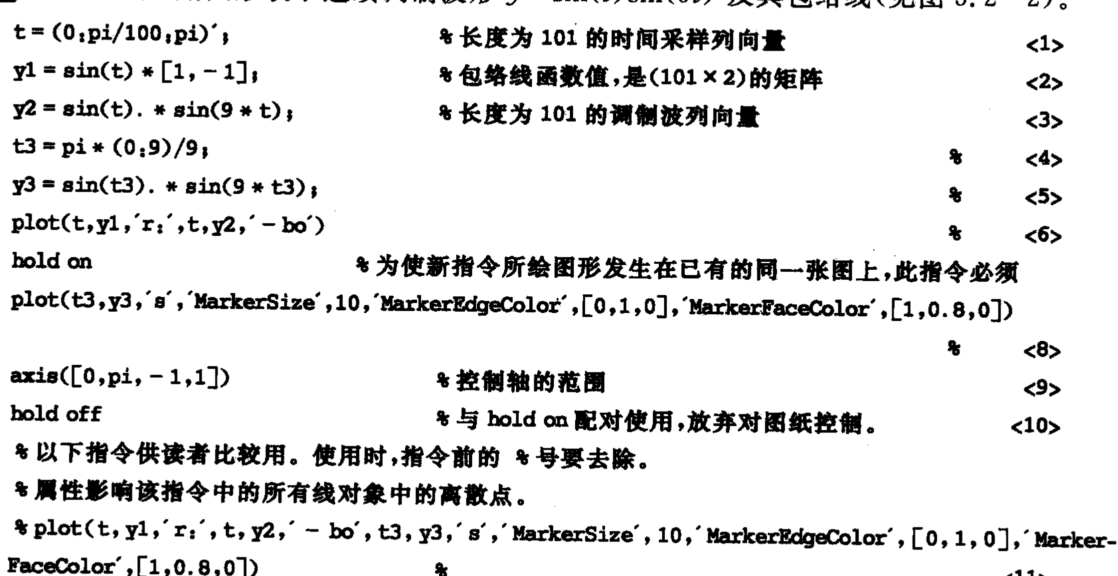


图 5.2-1　plot(t,Y)与plot(Y)所绘曲线的区别

说明

- 由于非彩色印刷，本例的色彩及其次序无法从印刷的书本上观察。请看光盘，或直接运行指令观察。
- plot(t,Y)与 plot(Y)所绘曲线的横坐标不同。

例【5.2-2】 用图形表示连续调制波形 $y=\sin(t)\sin(9t)$ 及其包络线(见图 5.2-2)。



```
t=(0:pi/100:pi)';                        % 长度为 101 的时间采样列向量          <1>
y1=sin(t)*[1,-1];                        % 包络线函数值,是(101×2)的矩阵         <2>
y2=sin(t).*sin(9*t);                     % 长度为 101 的调制波列向量            <3>
t3=pi*(0:9)/9;                                                          %      <4>
y3=sin(t3).*sin(9*t3);                                                  %      <5>
plot(t,y1,'r:',t,y2,'-bo')                                              %      <6>
hold on                      % 为使新指令所绘图形发生在已有的同一张图上,此指令必须
plot(t3,y3,'s','MarkerSize',10,'MarkerEdgeColor',[0,1,0],'MarkerFaceColor',[1,0.8,0])
                                                                        %      <8>
axis([0,pi,-1,1])                        % 控制轴的范围                         <9>
hold off                                 % 与 hold on 配对使用,放弃对图纸控制。 <10>
% 以下指令供读者比较用。使用时,指令前的 % 号要去除。
% 属性影响该指令中的所有线对象中的离散点。
% plot(t,y1,'r:',t,y2,'-bo',t3,y3,'s','MarkerSize',10,'MarkerEdgeColor',[0,1,0],'Marker-
FaceColor',[1,0.8,0])            %                                             <11>
```

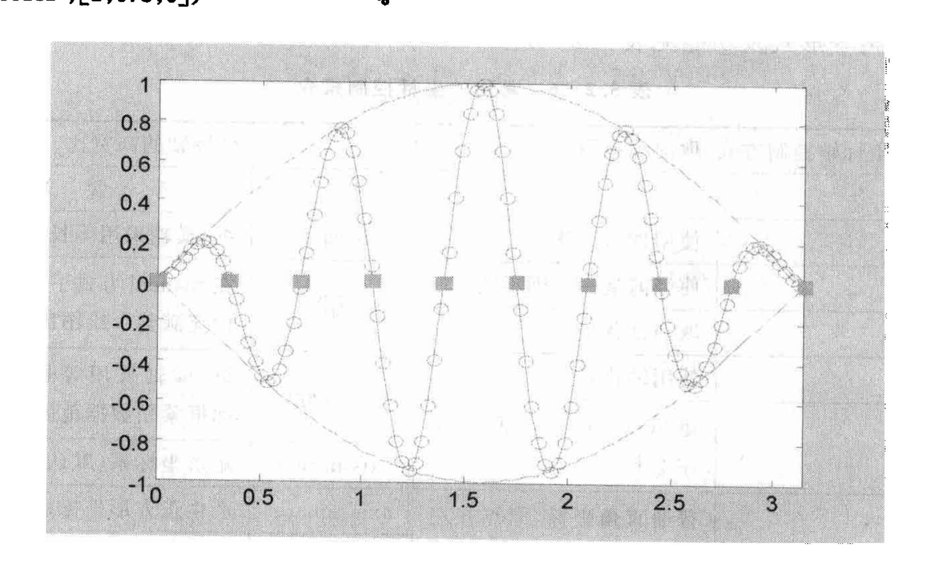


图 5.2-2　属性控制下所绘曲线

说明

- 产生自变量采样向量的方法很多。本例中的指令〈1〉,〈4〉是两种常用方法。本例指令〈1〉表示：自变量取值范围是$[0,\pi]$，采样间隔是 0.01π。
- 注意：指令〈1〉中，采用了转置操作。请读者尝试一下：假如在该指令中不用转置操作符，那么本例指令组运行结果是什么？如何修改其他指令，以得到图 5.2-2 的图形？

- 指令〈3〉和〈5〉中，采用了“数组乘”运算符。这大大简化和加速了函数值向量 $y2$ 的计算。这是 MATLAB 刻意设计数组运算的目的之一。如果不用“数组乘”，那么就不得不依靠费时低效的“循环”过程去计算函数值向量 $y2$。
- 指令〈6〉包含两个绘线“三元组”。第一个“三元组”画出两根红虚线。第二个“三元组”画一根带小圆点的蓝实线(小圆点标注数据点)。请注意指令书写方式。
- 由于曲线的过零点要进行较多的属性设置，为不与前面所画线对象冲突，而用另一条 plot 指令执行。
- 本例指令〈9〉设置了图形坐标轴的范围。请读者把最后的这条指令删去，观察运行后的图形有什么不同。
- 指令〈11〉提供给读者体验“属性设置”对所有曲线的影响。

5.2.2 坐标控制和图形标识

MATLAB 对图形风格的控制比较完备友善。一方面，在最通用的层面上，它采用了一系列考虑周全的默认设置，因此在绘制图形时，无需人工干预，它就能根据所给数据自动地确定坐标取向、范围、刻度、高宽比，并给出相当令人满意的画面。另一方面，在适应用户的层面上，它又给出了一系列便于使用的指令，可让用户根据需要和喜好改变默认设置。

1. 坐标轴的控制

坐标控制指令 axis 使用比较简单，它用于控制坐标轴的可视、取向、取值范围和轴的高宽比等。常用的指令形式及功能见表 5.2-6。

表 5.2-6 常用的坐标控制指令

坐标轴控制方式、取向和范围	
指 令	含 义
axis auto	使用默认设置
axis manual	使当前坐标范围不变
axis off	取消轴背景
axis on	使用轴背景
axis ij	矩阵式坐标，原点在左上方
axis xy	普通直角坐标，原点在左下方
axis(V) V=[x1,x2,y1,y2]; V=[x1,x2,y1,y2,z1,z2];	人工设定坐标范围。设定值：二维，4个；三维，6个

坐标轴的高宽比	
指 令	含 义
axis equal	纵、横轴采用等长刻度
axis fill	在 manual 方式下起作用，使坐标充满整个绘图区
axis image	纵、横轴采用等长刻度，且坐标框紧贴数据范围
axis normal	矩形坐标系(默认)
axis square	产生正方形坐标系
axis tight	把数据范围直接设为坐标范围
axis vis3d	保持高宽比不变，用于三维旋转时避免图形大小变化

说明

坐标范围设定向量 **V** 中的元素必须服从：$x1<x2, y1<y2, z1<z2$。**V** 的元素允许取 inf 或-inf，那意味着上限或下限是自动产生的，即坐标范围“半自动”确定。

【例 5.2-3】 观察各种轴控制指令的影响。演示采用长轴为 3.25，短轴为 1.15 的椭圆。注意：采用多子图(见图 5.2-3)表现时，图形形状不仅受控制指令影响，而且受整个图面宽高比及子图数目的影响。本书这样处理，是出于篇幅考虑。读者若想准确体会控制指令的影响，请在全图状态下进行观察。

```
t=0:2*pi/99:2*pi;
x=1.15*cos(t);y=3.25*sin(t);      %y为长轴,x为短轴
subplot(2,3,1),plot(x,y),axis normal,grid on,
title('Normal and Grid on')
subplot(2,3,2),plot(x,y),axis equal,grid on,title('Equal')
subplot(2,3,3),plot(x,y),axis square,grid on,title('Square')
subplot(2,3,4),plot(x,y),axis image,box off,title('Image and Box off')
subplot(2,3,5),plot(x,y),axis image fill,box off
title('Image and Fill')
subplot(2,3,6),plot(x,y),axis tight,box off,title('Tight')
```

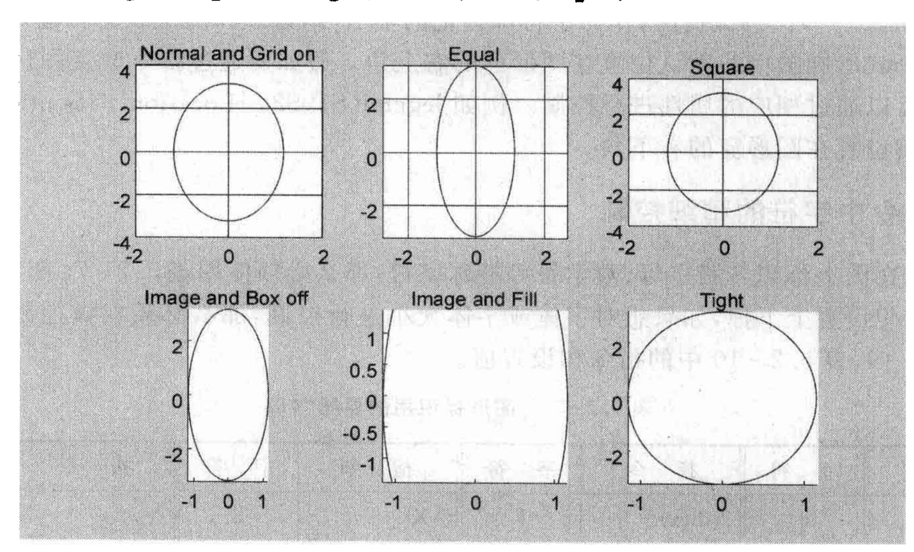


图 5.2-3 各种轴控制指令的不同影响

2. 分格线和坐标框

grid	是否画分格线的双向切换指令(使当前分格线状态翻转)
grid on	画出分格线
grid off	不画分格线
box	坐标形式在封闭式和开启式之间切换指令
box on	使当前坐标呈封闭形式
box off	使当前坐标呈开启形式

说明

- 不画分格线是 MATLAB 的默认设置。

● 分格线的疏密取决于坐标刻度。如想改变分格线的疏密，必须先定义坐标刻度。

● 默认情况下，所画坐标呈封闭形式。

3. 图形标识指令

图形标识包括：图名(title)、坐标轴名(label)、图形注释(text)和图例(legend)。标识指令的最简捷使用格式如下：

title(S)	书写图名
xlabel(S)	横坐标轴名
ylabel(S)	纵坐标轴名
legend(S1,S2, ...)	绘制曲线所用线型、色彩或数据点形图例
text(xt,yt,S)	在图面(xt, yt)坐标处书写字符注释

说明

● S，S1，S2 为字符串。它可以是英文、中文或 Tex 定义的各种特殊字符(见表 5.2-7～表 5.2-10)。再次提醒：作为字符串标记的单引号，必须在英文状态下输入。

● legend 所画图例的默认位置在图形窗的右上角。假如要通过指令改变图例的位置，那么可以通过相应的属性进行控制。例如 legend(S1,S2, 'Location', 'SouthEast')将把图例设置在图形窗的右下角。

4. 标识指令中字符的精细控制

如果想在图上标识希腊字母、数学符等特殊字符，那么必须使用表 5.2-7、表 5.2-8 中的指令。如果想设置上下标，如果想对字体或字体大小进行控制，那么必须在被控制字符前，先使用表 5.2-9、表 5.2-10 中的指令和设置值。

表 5.2-7 图形标识用的希腊字母

指 令	字 符	指 令	字 符	指 令	字 符	指 令	字 符
\alpha	α	\theta	θ	\Xi	Ξ	\phi	φ
\beta	β	\Theta	Θ	\pi	π	\Phi	Φ
\gamma	γ	\iota	ι	\Pi	Π	\chi	χ
\Gamma	Γ	\kappa	κ	\rho	ρ	\psi	ψ
\delta	δ	\lambda	λ	\sigma	σ	\Psi	Ψ
\Delta	Δ	\Lambda	Λ	\Sigma	Σ	\omega	ω
\epsilon	ε	\mu	μ	\tau	τ	\Omega	Ω
\zeta	ζ	\Nu	ν	\upsilon	υ		
\eta	η	\xi	ξ	\Upsilon	Y		
使用示例							
指 令	效 果	指 令	效果	指 令			效果
'sin\beta'	$\sin\beta$	'\zeta\omega'	$\zeta\omega$	'\itA{\in}R^{m\timesn}'			$A\in R^{m\times n}$

表 5.2-8　图形标识用的其他特殊字符

指　令	符	指　令	符	指　令	符	指　令	符	指　令	符
\approx	$\approx$	\propto	$\propto$	\exists	$\exists$	\cap	$\cap$	\downarrow	$\downarrow$
\cong	$\cong$	\sim	$\sim$	\forall	$\forall$	\cup	$\cup$	\leftarrow	$\leftarrow$
\div	$\div$	\times	$\times$	\in	$\in$	\subset	$\subset$	\leftrightarrow	$\leftrightarrow$
\equiv	$\equiv$	\oplus	$\oplus$	\infty	∞	\subseteq	$\subseteq$	\rightarrow	$\rightarrow$
\geq	$\geqslant$	\oslash	$\oslash$	\perp	$\perp$	\supset	$\supset$	\uparrow	$\uparrow$
\leq	$\leqslant$	\otimes	$\otimes$	\prime	$'$	\supseteq	$\supseteq$	\circ	$\circ$
\neq	$\neq$	\int	$\int$	\cdot	$\cdot$	\Im	$\Im$	\bullet	$\bullet$
\pm	$\pm$	\partial	∂	\ldots	$\ldots$	\Re	$\Re$	\copyright	©

表 5.2-9　上下标的控制指令

分　类	指　令	arg 取值	举　例	
			示例指令	效　果
上　标	^{arg}	任何合法字符	'\ite^{-t}sint'	$e^{-t}\sin t$
下　标	_{arg}	任何合法字符	'x~{\chi}_{\alpha}^{2}(3)'	$x \sim \chi_{\alpha}^{2}(3)$

表 5.2-10　字体式样设置规则

字　体	指　令	arg 取　值	举　例	
			示例指令	效　果
名　称	\fontname{arg}	arial; courier; roman; 宋体;隶书;黑体……	'\fontname{courier}Example 1' '\fontname{隶书}范例 2'	Example 1 范例 2
风　格	\arg	bf （黑体） it （斜体一） sl （斜体二） rm （正体）	'\bfExample 3' '\itExample 4'	**Example 3** *Example 4*
大　小	\fontsize{arg}	正整数。 默认值为 10 (Points 磅)。	'\fontsize{14}Example 5' '\fontsize{6}Example 6'	Example 5 Example 6

说明

- 凡 Windows 字库中有的字体，都可以通过设置字体名称实现调用。
- 对中文进行字体选择是允许的。见例 5.2-4 的第<5>条指令。
- Notebook 和指令窗之间交叉操作时，中文字体的设置有时会引起图形注释混乱。
- 1 Point(磅) = 1/72 inch = 0.35 mm。

例【5.2-4】　本例非常简单，专供试验标识用(见图 5.2-4)。读者在指令窗中反复调用

这两条指令就可以检查自己对指令、标识的理解是否正确。当然每次试验时，第<5>条指令中的字符串应按读者的需要改变。

```
clf;t = 0:pi/50:2 * pi;
y = sin(t);
plot(t,y)
axis([0,2 * pi, - 1.2,1.2])
text(pi/2,1,'\fontsize{16}\leftarrow\itsin(t)\fontname{隶书}极大值')
                                                                    % <5>
title('y = sin(t)')
xlabel('t')
ylabel('y')
```

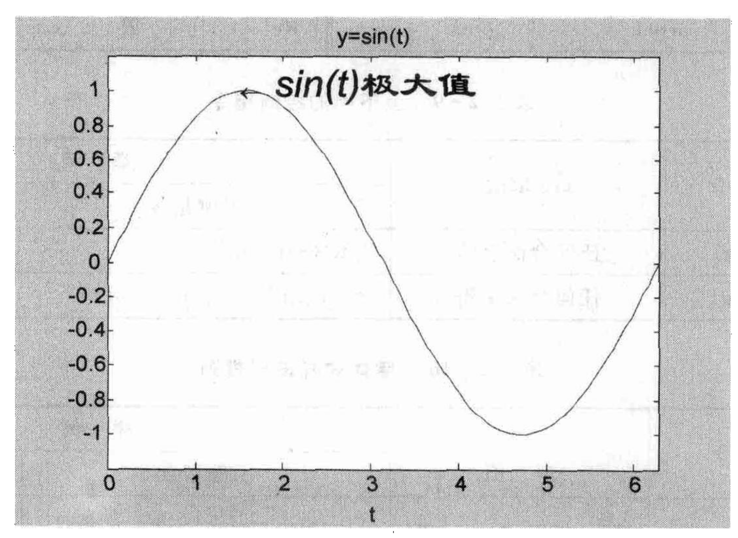


图 5.2-4　试验标识的图形

例【5.2-5】 通过绘制二阶系统阶跃响应，综合演示图形标识。本例比较综合，涉及的指令较广。请读者耐心读，实际做，再看例后说明，定会有匪浅收益（见图 5.2-5）。

```
clf;t = 6 * pi * (0:100)/100;
y = 1 - exp( - 0.3 * t). * cos(0.7 * t);
plot(t,y,'r - ','LineWidth',3)          % 用 3 磅红实线画曲线              <3>
hold on
tt = t(find(abs(y - 1)>0.05));ts = max(tt);                               % <5>
plot(ts,0.95,'bo','MarkerSize',10)      % 镇定点位置                        <6>
hold off
axis([ - inf,6 * pi,0.6,inf])           % 横坐标下限及纵坐标上限自动生成
set(gca,'Xtick',[2 * pi,4 * pi,6 * pi],'Ytick',[0.95,1,1.05,max(y)])      % <9>
set(gca,'XtickLabel',{'2 * pi';'4 * pi';'6 * pi'})                       % <10>
set(gca,'YtickLabel',{'0.95';'1';'1.05';'max(y)'})                       % <11>
grid on
text(13.5,1.2,'\fontsize{12}{\alpha} = 0.3')                              % <13>
```

```
text(13.5,1.1,'\fontsize{12}{\omega} = 0.7')                              % <14>

cell_string{1} = '\fontsize{12}\uparrow';                                  % <15>
cell_string{2} = '\fontsize{16} \fontname{隶书}镇定时间';
cell_string{3} = '\fontsize{6}  ';
cell_string{4} = ['\fontsize{14}\rmt_{s} = ' num2str(ts)];                 % <18>
text(ts,0.85,cell_string,'Color','b','HorizontalAlignment','Center')
                                                                           % <19>
title('\fontsize{14}\it y = 1 - e^{ -\alpha t}cos{\omegat}')               % <20>
xlabel('\fontsize{14} \bft \rightarrow')      % 横坐标标识
ylabel('\fontsize{14} \bfy \rightarrow')      % 纵坐标标识                  <22>
```

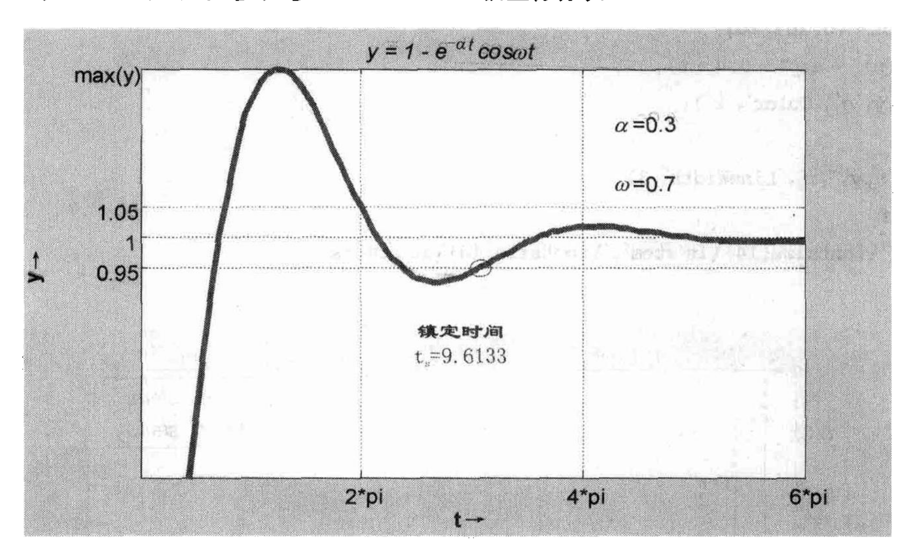


图 5.2-5　二阶阶跃响应图的标识

说明

- 指令<5>旨在寻找镇定时间。即从此以后，响应与 1 的距离再也不会超过 0.05。
- 指令<9>设置坐标轴上的分度线位置。
- 指令<10><11>通过手工设置，分别对 x，y 轴的分度线进行标识。x 轴采用 π 的偶倍数标识，以增强可读性。注意指令中的“花括号及其中的分号”。
- 指令<13><14>采用 12 磅字体书写 $\alpha=0.3$ 和 $\omega=0.7$。
- 指令<15>到<18>构成一个字符串“胞元(Cell)”数组。它将用于指令<19>的“多行注释”。
- 指令<19>对镇定点采用“蓝色”和“中心对准”的方式进行“多行注释”。
- 指令<20>使用“斜体”、“希腊字符”、“上标”和 14 磅字体书写图名。
- 指令<22>中，注意使用的是**rightarrow**，而不是**uparrow**。

5.2.3 多次叠绘、双纵坐标和多子图

1. 多次叠绘

第 5.2.1 节已经介绍过 plot 在同一次调用中画多条曲线的功能。实际应用中，还会遇到在已经存在的图上再绘制一条或多条曲线的情况。为此，MATLAB 提供了以下指令。

hold on	使当前轴及图形保持而不被刷新，准备接受此后将绘制的新曲线
hold off	使当前轴及图形不再具备不被刷新的性质
hold	当前图形是否具备刷新性质的双向切换开关

【例 5.2-6】 利用 hold 绘制离散信号通过零阶保持器后产生的波形（见图 5.2-6）。

```
t=2*pi*(0:20)/20;
y=cos(t).*exp(-0.4*t);
stem(t,y,'g','Color','k');
hold on
stairs(t,y,':r','LineWidth',3)
hold off
legend('\fontsize{14}\it stem','\fontsize{14}\it stairs')
box on
```

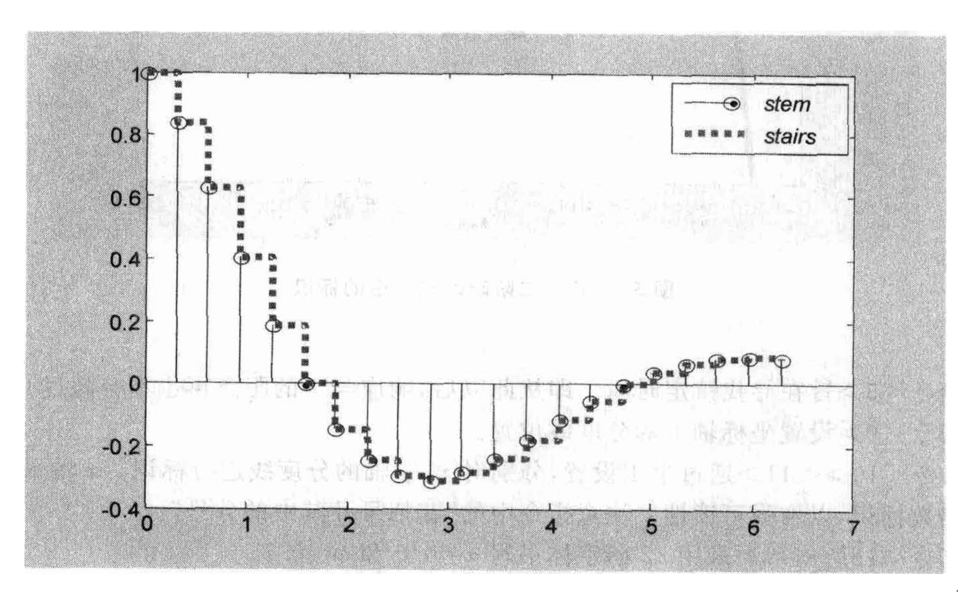


图 5.2-6 离散信号的重构

说明

指令 hold 用得较多。如例 5.2-5 中的指令<4>等。事实上，如想在同一图形窗里，不是一次使用如 plot，stem，stairs 等绘图指令，那么就必须使用 hold。

2. 双纵坐标图

在实际应用中常常提出这样一种需求：把同一自变量的两个不同量纲、不同数量级的函数

量的变化绘制在同一张图上。如希望在同一张图上表现温度、湿度随时间的变化；温度、压力的响应曲线；人口数量、GDP 的变化曲线；放大器输入、输出电流变化曲线等。为满足这种需求，MATLAB 提供了以下指令。

plotyy(X1,Y1,X2,Y2)

以左、右不同纵轴绘制 $X1-Y1$、$X2-Y2$ 两条曲线

plotyy(X1,Y1,X2,Y2,'FUN')

以左、右不同纵轴把 $X1-Y1$、$X2-Y2$ 绘制成 FUN 指定形式的两条曲线

plotyy(X1,Y1,X2,Y2,'FUN1','FUN2')

以左、右不同纵轴把 $X1-Y1$、$X2-Y2$ 绘制成 FUN1、FUN2 指定的不同形式的两条曲线

说明

- 左纵轴用于 $X1-Y1$ 数据对，右纵轴用于 $X2-Y2$ 数据对。
- 轴的范围、刻度都自动产生。如果想人工设置，那必须使用图柄和低层绘图指令。
- FUN、FUN1、FUN2 可以是 MATLAB 中所有接收 $X-Y$ 数据对的二维绘图指令。

例【5.2-7】 画出函数 $y=x\sin x$ 和积分 $s=\int_0^x(x\sin x)dx$ 在区间[0,4]上的曲线（见图 5.2-7）。

```
clf;dx = 0.1;x = 0:dx:4;y = x.*sin(x);
s = cumtrapz(y)*dx;                               % 梯形法求累计积分        <2>
a = plotyy(x,y,x,s,'stem','plot');                                       % <3>
text(0.5,1.5,'\fontsize{14}\ity = xsinx')         % 对被积函数加注          <4>
sint = '{\fontsize{16}\int_{\fontsize{8}0}^{ x}}';                       % <5>
ss = ['\fontsize{14}\its = ',sint,'\fontsize{14}\itxsinxdx'];            % <6>
text(2.5,3.5,ss)                                   % 对积分得到的原函数加注  <7>
set(get(a(1),'Ylabel'),'String','被积函数 \ity = xsinx')                 % <8>
set(get(a(2),'Ylabel'),'String',ss)                                      % <9>
xlabel('x')
```

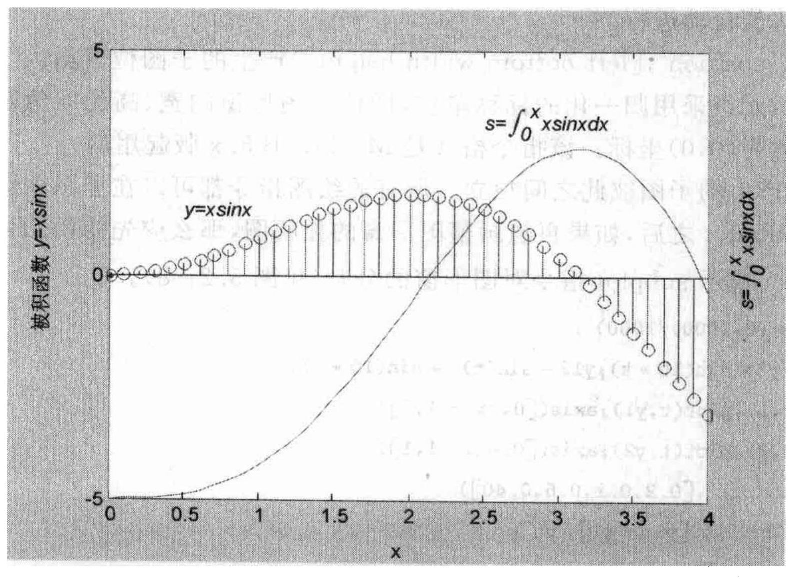


图 5.2-7　函数和积分

说明

- 指令＜2＞中，**cumtrapz** 用于求累计积分(见第 4.1.2 节)。
- 指令＜3＞分别采用"杆图"、"线图"绘制被积函数、原函数，并把重合在一起的两张图的"轴对象句柄"赋给 a，以供指令＜8＞＜9＞使用。
- 指令＜4＞对被积函数的图形加注。
- 指令＜5＞＜6＞用于合成供指令＜7＞使用的 $s=\int_0^x x\sin x\mathrm{d}x$ 字符串。这样做的目的是，避免一条指令写得太长。
- 关于指令＜7＞，要特别说明：在 plotyy 生成的图形中，使用 text 指令加注标识文字的位置是根据左纵轴决定的。
- 指令＜8＞中的 a(1)是包含左纵轴的那个"轴对象句柄"；get(a(1),'Ylabel') 用来获得那轴上的"纵坐标名句柄"。通过该指令＜7＞把左纵轴名称写成"被积函数 $y=x\sin x$"。
- 指令＜9＞用于右纵轴名的书写。
- **legend** 指令不能正常使用于此双纵坐标。
- 本例使用到的图形修饰指令很有特色，有兴趣的读者可仔细阅读。

3. 多子图

MATLAB 允许用户在同一个图形窗里布置几幅独立的子图。具体指令是：

subplot(m,n,k)	使($m\times n$)幅子图中的第 k 幅成为当前图
subplot('position',[left bottom width height])	在指定位置上开辟子图，并成为当前图

说明

- subplot(m,n,k)的含义是：图形窗中将有($m\times n$)幅子图。k 是子图的编号。子图的序号编排原则是：左上方为第 1 幅，向右向下依次排号。该指令形式产生的子图分割完全按默认值自动进行。
- subplot('position',[left bottom width height])产生的子图位置由人工指定。那指定位置的四元组采用归一化的标称单位，即认为图形窗的宽、高的取值范围都是[0,1]，而左下角为(0,0)坐标。该指令格式是 MATLAB 5.x 版起用的。
- subplot 产生的子图彼此之间独立。所有的绘图指令都可以在子图中运用。
- 在使用**subplot** 之后，如果再想画整图形窗的独幅图，那么应先使用**clf** 指令清图形窗。

例【5.2-8】 演示 subplot 指令对图形窗的分割(见图 5.2-8)。

```
clf;t = (pi * (0:1000)/1000)';
y1 = sin(t);y2 = sin(10 * t);y12 = sin(t). * sin(10 * t);
subplot(2,2,1),plot(t,y1);axis([0,pi, - 1,1])
subplot(2,2,2),plot(t,y2);axis([0,pi, - 1,1])
subplot('position',[0.2,0.1,0.6,0.40])
plot(t,y12,'b- ',t,[y1, - y1],'r:')
axis([0,pi, - 1,1])
```

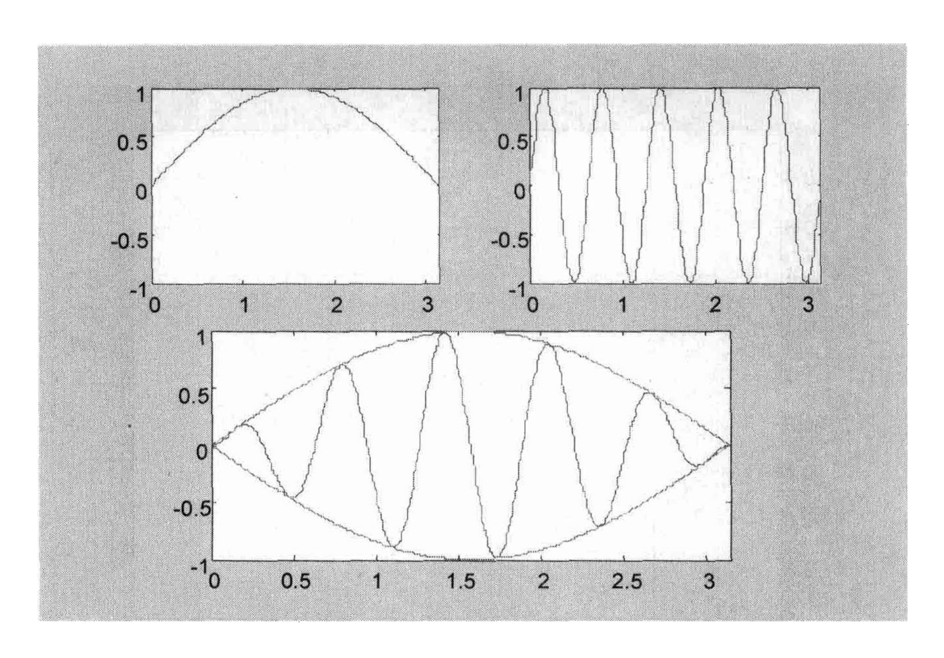


图 5.2-8　多子图的布置

5.2.4　获取二维图形数据的指令 ginput

[x,y]=ginput(n)　　　用鼠标从二维图形上获取 n 个点的数据坐标 (x,y) 。

说明

- 该指令能从图形获取数据的指令。该指令在数值优化、方程求解、及工程设计中十分有用。
- 该指令仅适用于二维图形。指令中的 n 应赋正整数，它是用户希望通过鼠标从图上获得的数据点数目。指令中的 x，y 存放所取点的坐标。
- 该指令具体操作方法：该指令运行后，会把当前图形从后台调到前台，同时鼠标光标变化为十字叉；用户可移动鼠标，使十字叉移到待取坐标点；点动鼠标左键，便获得该点数据；此后，用同样的方法，获取其余点的数据；当 n 个点的数据全部取到后，图形窗便退回后台，机器回到 ginput 执行前的环境。
- 在使用该指令之前，通常先对图形进行局部放大处理。

例【5.2-9】 采用图解法求 $(x+2)^x=2$ 的解。

(1) 绘制 $y=(x+2)^x-2$ 的曲线

曲线绘制区间采用尝试法确定。先根据方程，大致选择一个自变量范围，比如 [−1, 5]。然后绘制曲线，对图形观察后，可以进一步选择更恰当的自变量区间，比如 [0, 1]，再绘制出如图 5.2-9 所示的图形。

```
clf
x=0:0.01:1;
y=(x+2).^x-2;
plot(x,y);
```

```
grid on
```

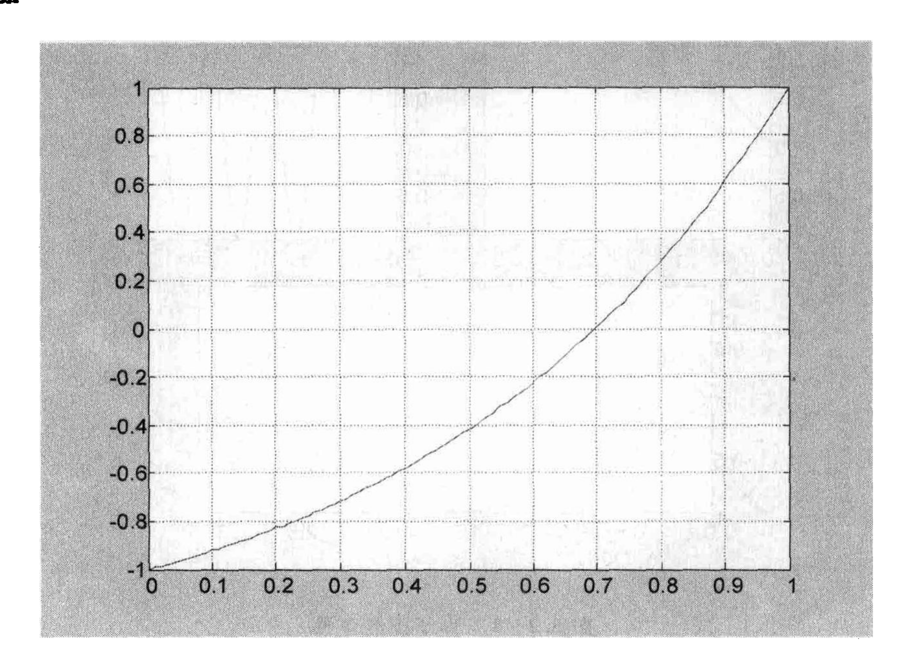


图 5.2－9　经多次试探后画出的图形

（2）在曲线与坐标横轴（y ＝ 0）的交点附近局部放大图（见图 5.2－10）上求解

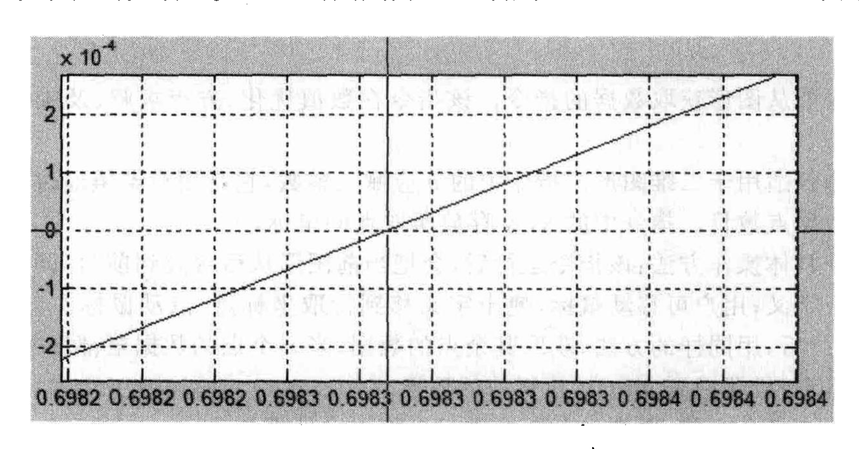


图 5.2－10　局部放大图

利用图形窗上的放大工具图标，把交点 y ＝ 0 附近的区域放到足够大，如图 5.2－10 所示。在该图上，观察横坐标分隔线及其数字标识，可以发现：同一个数字标识了相邻的几条不同分隔线。这表明：横坐标再进一步放大已经没有用什么意义了，换句话说，横坐标已经放得足够大了。在此，只关心横坐标放得足够大，是因为"要求取得交点横坐标"。

在把图形放得足够大以后，再运行以下指令，以便用鼠标获取交点坐标的数值。

```
[x,y] = ginput(1);
```

(3) 显示图解结果

```
format long
x,y            % 本题在例 2.6-6 中的符号解为 0.6982994217…
x =
   0.69828692903537
y =
   -5.884401711382421e-010
```

说明

- 本例图解的精度达到万分之一。
- 图解精度与自变量采样点密度、局部放大倍数、鼠标取值操作有关。

5.3　三维曲线和曲面

5.3.1　三维线图指令 plot3

plot3(X,Y,Z,'s')	用 s 指定的点形线型色彩绘制曲线
plot3(X1,Y1,Z1,'s1',X2,Y2,Z2,'s2', ...)	用 s1, s2 指定的点形线型色彩绘制多类曲线

说明

- plot3 的使用方法与 plot 相似。
- ***X***、***Y***、***Z*** 是同维向量时，则绘制以 ***X***、***Y***、***Z*** 元素为 x,y,z 坐标的三维曲线。
- ***X***、***Y***、***Z*** 是同维矩阵时，则以 ***X***、***Y***、***Z*** 对应列元素为 x,y,z 坐标分别绘制曲线，曲线条数等于矩阵的列数。
- s ，s1，s2 的意义与二维情况完全相同。它们用来指定色彩、线型、数据点形的选项字符串。它们的合法取值请看第 5.2.1 节。它们可以空缺，那时线型、色彩将由 MATLAB 的默认设置确定。
- 绘线“四元组”(X1,Y1,Z1,'s1')、(X2,Y2,Z2,'s2')的结构和作用，与(X,Y,Z,'s')相同。不同“四元组”之间没有约束关系。
- 三维线图指令 plot3 主要用来表现单参数的三维曲线。

例【5.3-1】　三维曲线绘图。本例演示：三维曲线的参数方程；线型、点形和图例（见图 5.3-1）。

```
t=(0:0.02:2)*pi;
x=sin(t);y=cos(t);z=cos(2*t);          % 三维曲线的参数方程
plot3(x,y,z,'b-',x,y,z,'bd')
view([-82,58]),box on
xlabel('x'),ylabel('y'),zlabel('z')
legend('链','宝石','Location','best')
```

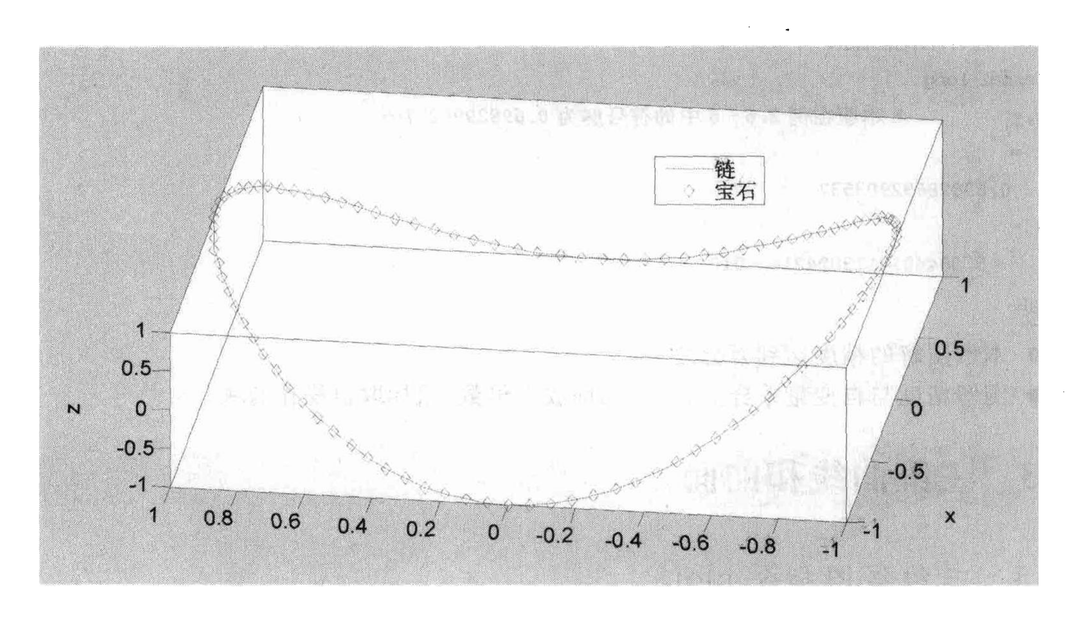


图 5.3-1 宝石项链

5.3.2 三维曲面/网线图

三维网线图和曲面图的绘制比线图稍显复杂。这主要表现于:绘图数据的准备;三维图形的色彩、明暗、光照和视点处理。

1. 三维曲面/网线可视化的数据准备

画函数 $z=f(x,y)$所代表的三维空间曲面,需要做以下数据准备。

(1) 确定自变量 x,y 的取值范围和取值间隔。x=x1:dx:x2;y=y1:dy:y2;

(2) 构成 $x-y$ 平面上的自变量"格点"矩阵。[X,Y]=meshgrid(x,y);

(3) 计算在自变量采样"格点"上的函数值。Z=f(X,Y)

2. 绘制曲面/网线图的基本指令

surf(Z)	以 Z 矩阵列、行下标为 x,y 轴自变量,画曲面图
surf(X,Y,Z)	最常用的曲面图调用格式
surf(X,Y,Z,C)	最完整调用格式,画由 C 指定用色的曲面图
mesh(Z)	以 Z 矩阵列、行下标为 x,y 轴自变量,画网线图
mesh(X,Y,Z)	最常用的网线图调用格式
mesh(X,Y,Z,C)	最完整调用格式,画由 C 指定用色的网线图

说明

- 在最完整调用格式中的四个输入宗量都是维数相同的矩阵。***X***,***Y*** 是自变量"格点"矩阵;***Z*** 是格点上的函数矩阵;***C*** 是指定各点用色的矩阵。***C*** 可以空缺,此时,默认用色矩阵是 ***Z***,即认为 ***C***=***Z***。
- 单输入宗量格式绘图时,把 ***Z*** 矩阵的列下标当作 x 坐标轴的"自变量",把 ***Z*** 的行下标

当作 y 坐标轴的“自变量”。

例【5.3-2】 用曲面图表现函数 $z=x^2+y^2$。本例演示：三维空间中曲面图的绘制步骤和成图原理；x-y 平面上的采样数据“格点”(见图 5.3-2)。

```
clf
x= -4:4;y=x;                    % 确定自变量的取值范围和采样间隔
[X,Y]=meshgrid(x,y);            % 生成 x-y 平面上的自变量“格点”矩阵
Z=X.^2+Y.^2;                    % 计算格点上的函数值
surf(X,Y,Z);                    % 绘制曲面图
colormap(hot)                   % 采用 hot 色图
hold on
stem3(X,Y,Z,'bo')               % 用来表现 x-y 平面格点,及对应的函数值
hold off
xlabel('x'),ylabel('y'),zlabel('z')
axis([-5,5,-5,5,0,inf])         % 设置坐标范围
view([-84,21])                  % 控制视角
```

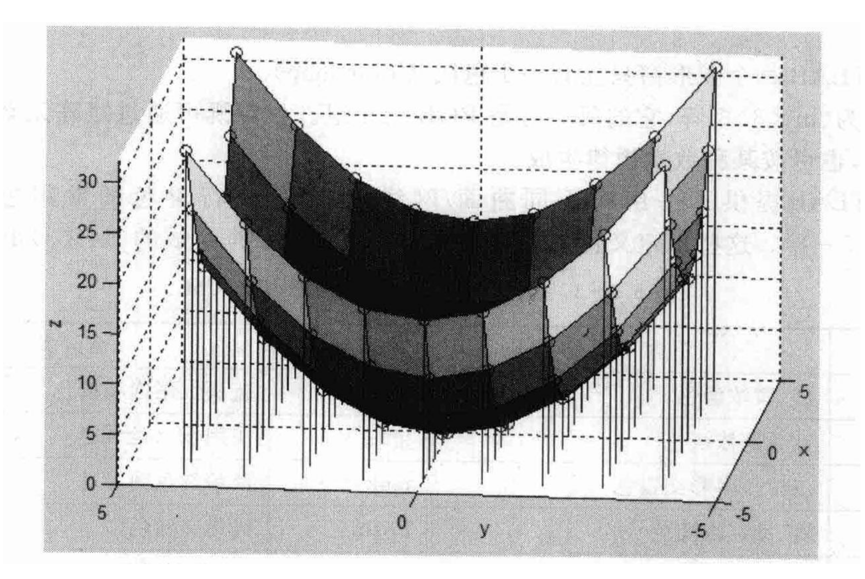


图 5.3-2 曲面图和格点

5.3.3 曲面/网线图的精细修饰

1. 视角控制 view

视点设置参数方法如下，其示意图见图 5.3-3。

view([az,el]) 通过方位角、俯视角设置视点

view([vx,vy,vz]) 通过直角坐标设置视点

说明

- 改变观察点是获得较好三维视觉效果的一个重要途径。
- 指令中，az 是方位角(Azimuth)，el 是俯视角(Elevation)。它们的单位是“度”。vx，

vy, *vz* 是视点的直角坐标。

- 倘若绘制三维图形时，不使用 view 指令，那么 MATLAB将使用视点默认设置：$az=-37.5°$，$el=30°$。当 $az=0°$，$el=90°$时，图形将以习惯的平面直角坐标表现。
- MATLAB 图形窗可交互式调节视点。为获得最佳视觉效果，用户先通过鼠标操作调节视点，然后用 [az0, el0]=view 获得满意视觉效果时的方位角和俯视角，再把这对参数通过 view([az0,el0])用于绘图的一组指令中。（注意：有时指令 [az0, el0]=view 给出的 el0 数据比真实的俯视角大 30°左右。）

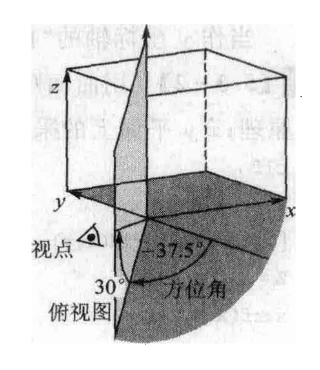


图 5.3-3 视点设置参数示意

2. 色图 colormap

colormap(CM) 设置当前图形窗的着色色图为 CM

说明

- MATLAB 每个图形窗只能有一个色图(Color map)。
- 色图为(m×3)矩阵，它的每一行是 RGB 三元组。色像既可通过矩阵元素的直接赋值定义，也可按某种数学规律生成。
- MATLAB 提供了一组对不同曲面/网线图、图像进行着色的常用色图矩阵（见表 5.3-1）。这些预定义色图矩阵是由 [0, 1]区间数据组成的(64×3)矩阵。

表 5.3-1 MATLAB 的预定义色图矩阵 CM

CM	含 义	CM	含 义
autumn	红、黄浓淡色	jet	蓝头红尾饱和值色
bone	蓝色调浓淡色	lines	采用 plot 绘线色
colorcube	三浓淡多彩交错色	pink	淡粉红色图
cool	青、品红浓淡色	prism	光谱交错色
copper	纯铜色调线性浓淡色	spring	青、黄浓淡色
flag	红-白-蓝-黑交错色	summer	绿、黄浓淡色
gray	灰色调线性浓淡色	winter	蓝、绿浓淡色
hot	黑、红、黄、白浓淡色	white	全白色
hsv	两端为红的饱和值色		

注：jet 是默认色图。

3. 浓淡处理 shading

shading options 图形对象着色的浓淡处理

说明

- mesh 、surf 、pcolor 、fill 和 fill3 所创建图形非数据点处的着色由 shading 指令决定。
- 指令的选项options 可取以下三种方式。

flat	网线图的某整条线段，或曲面图的某整个贴片都着一种颜色；颜色取自该线段两端，或该贴片四顶点数据点中下标最小那点的颜色。
interp	网线图线段，或曲面图贴片上各点的颜色由该线段两端，或该贴片四顶点处的颜色经线性插值而得；这种方法的用色比较细腻，但最费时。
faceted	在 flat 用色基础上，再在贴片的四周勾画黑色网线；这种方法对立体的表现力最强，因此 MATLAB 把它作为默认设置。

- shading 是设置当前轴上"面"对象的 EdgeColor 和 FaceColor 属性的高层指令。

例【5.3-3】 三种浓淡处理方式比较。本例演示：shading 的功用；图形窗底色的设置（见图 5.3-4）。

```
clf                                          % 清空图形窗
x = -4:4;y = x;
[X,Y] = meshgrid(x,y);
Z = X.^2 + Y.^2;
surf(X,Y,Z)
colormap(jet)
subplot(1,3,1),surf(Z),axis off
subplot(1,3,2),surf(Z),axis off,shading flat
subplot(1,3,3),surf(Z),axis off,shading interp
set(gcf,'Color','w')               % 设置图形窗的底色为白
```

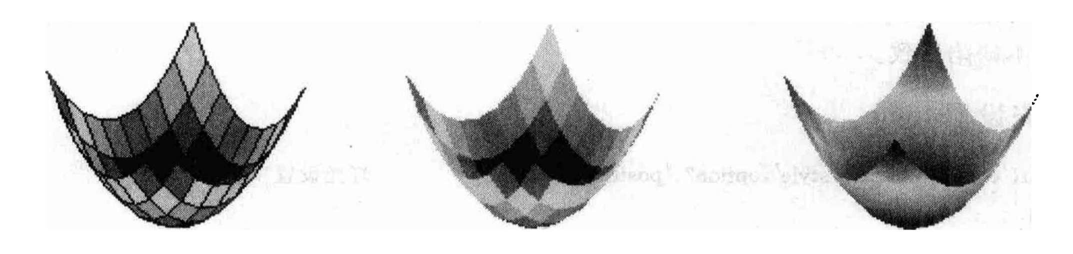


图 5.3-4　三种浓淡处理方式比较

说明

运行set(gcf, 'Color', 'default')，就可恢复图形窗的默认底色。

4. 透明度控制 alpha

alpha(v)	对面、块、象三种图形对象的透明度加以控制

说明

- v 可以取 0 到 1 之间的数值。0 表示完全透明，1 表示不透明。
- 本指令对 mesh，surf，slice 等高层指令都适用。

例【5.3-4】 半透明的表面图。本例演示：alpha 指令；MATLAB 提供的双变量正态分布

曲面函数的用法(见图 5.3-5)。

```
clf
surf(peaks)
shading interp
alpha(0.5)
colormap(summer)
```

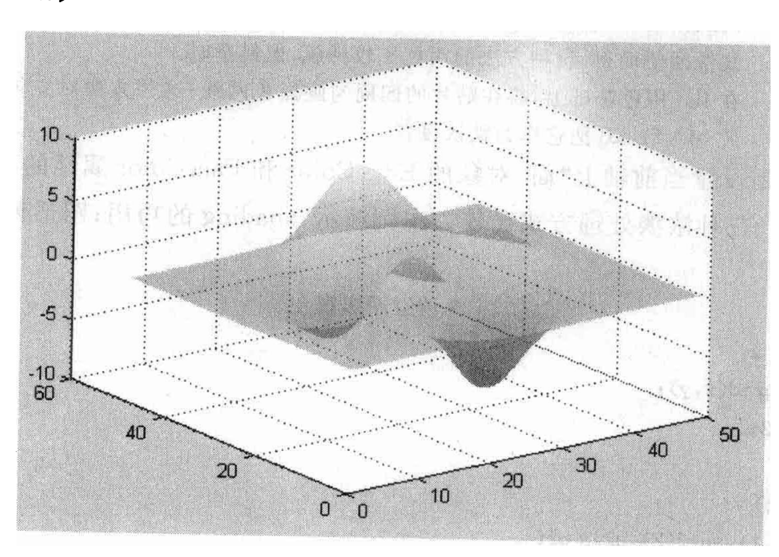


图 5.3-5 半透明薄膜

说明

值得指出:在 MATLAB R2010a 的 Notebook 环境中,alpha 指令有可能无法控制透明度。这是版本缺陷所致。

5. 灯光设置 light

light('color',option1,'style',option2,'position',option3)　　灯光设置

说明

- 在该指令使用前,图形采用强度各处相等的漫射光。一旦该指令被执行,虽然光源本身并不出现,但图形上"面"等子对象所有与"光"有关的属性(如背景光,边缘光)都被激活。
- 假如该指令不包含任何输入宗量,则采用默认设置:白光、无穷远、穿过[1, 0, 1]射向坐标原点。
- 任何一个输入宗量对都可以空缺,那时采用默认设置。
- option1 可采用 RGB 三元组或相应的色彩字符。如[1 0 0],或'r',都代表红光。
- option2 有两个取值:'infinite' 和 'local'。前者表示无穷远光,后者为近光。
- options3 总为直角坐标的三元数组形式。对远光,它表示光线穿过该点射向原点;对近光,它表示光源所在位置。

6. 照明模式 lighting

lighting　options　　　　设置照明模式

说明

- 该指令只有在 light 指令执行后才起作用。
- options 有以下四种取值。

flat　　入射光均匀洒落在图形对象的每个面上。主要与 facted 配用；它是默认模式。
gouraud　　先对顶点颜色插补，再对顶点勾画的面色进行插补；用于曲面表现。
phong　　对顶点处法线插值，再计算各象素的反光；表现效果最好，但费时较多。
none　　使所有光源关闭。

7. 控制光反射的材质指令 material

material options　　　　使用预定义反射模式

说明

为用户使用方便，MATLAB 提供了以下四种预定义表面反射模式，即options 的取值。

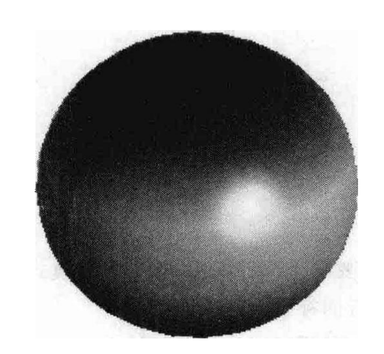
shiny　　使对象比较明亮；镜反射份额较大，反射光颜色仅取决于光源颜色。
dull　　使对象比较暗淡；漫反射份额较大，没有镜面亮点，反射光颜色仅取决于光源颜色。
metal　　使对象带金属光泽；镜反射份额很大，背景光和漫反射份额很小；反射光颜色取决于光源和图形表面两者的颜色；该模式为默认设置。
default　　返回默认设置模式。

例【5.3-5】 灯光、照明、材质指令所表现的图形(见图 5.3-6)。

```
clf;
[X,Y,Z] = sphere(40);
colormap(jet)                                                   % <3>
subplot(1,2,1),surf(X,Y,Z),axis equal off,shading interp        % <4>
light ('position',[0 -10 1.5],'style','infinite')               % <5>
lighting  phong                                                 % <6>
material shiny                                                  % <7>
subplot(1,2,2),surf(X,Y,Z,-Z),axis equal off,shading flat       % <8>
light;lighting flat                                             % <9>
light('position',[-1,-1,-2],'color','y')                        % <10>
light('position',[-1,0.5,1],'style','local','color','w')        % <11>
set(gcf,'Color','w')          % 设置图形窗的底色为白
```

说明

- 色图是图形窗的属性。每个图形窗只有一个色图。见本例指令〈3〉。
- 每个子图可以定义自己的浓淡处理模式、照明模式、材质，但它们都只能定义一次。如本例左子图，相关定义指令是〈4〉～〈7〉；而右子图，相关定义指令是〈8〉～〈11〉。
- 每个子图上可以设置多个光源。本例左子图，只使用了 1 个默认设置光源。而图 5.3-6 右子图使用了包括默认设置光源在内的 3 个形式、方向、颜色不同的光源。



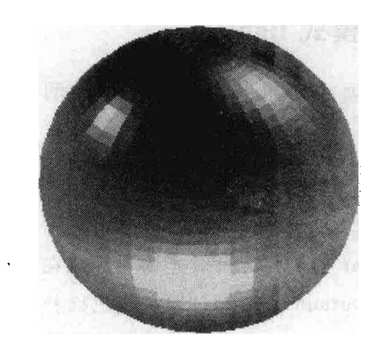


图 5.3-6 灯光、照明、材质指令所表现的图形

5.3.4 透视、镂空和裁切

1. 图形的透视

MATLAB在采用默认设置画 mesh 图形时，对叠压在后面的图形采取了消隐措施。但有时却需要透视效果。为此，MATLAB提供了一个控制消隐的指令如下

hidden off 透视被叠压的图形

hidden on 消隐被叠压的图形

例【5.3-6】 透视演示(见图 5.3-7)。

```
[X0,Y0,Z0] = sphere(30);           % 产生单位球面的三维坐标
X = 2 * X0;Y = 2 * Y0;Z = 2 * Z0;   % 产生半径为 2 的球面的三维坐标
surf(X0,Y0,Z0);                    % 画单位球面
shading interp                     % 采用插补明暗处理
```

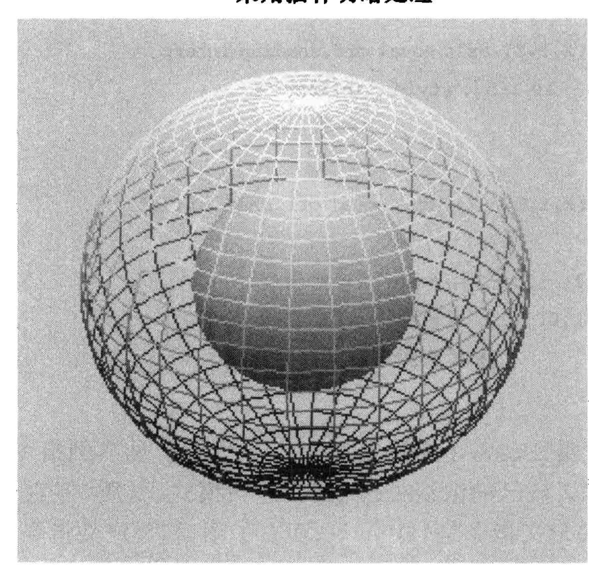


图 5.3-7 剔透玲珑球

```
hold on,mesh(X,Y,Z),colormap(hot)     % 采用 hot 色图
hold off
hidden off                            % 产生透视效果
axis equal,axis off                   % 不显示坐标轴
```

说明

- hidden 指令对surf 指令绘制的表面图不发生任何影响。
- alpha 指令可对透明度进行更细致的控制。

2. 图形的镂空

例【5.3-7】 演示：如何利用"非数"NaN，对图形进行镂空处理(见图 5.3-8)。

```
P = peaks(30);                      % 从 MATLAB 提供的双变量正态分布曲面获取数据
P(18:20,9:15) = NaN;                % 为镂空而赋值
surfc(P);                           % 带等位线的曲面图
colormap(hot)
light('position',[50, - 10,5])
material([0.9,0.9,0.6,15,0.4])
grid off,box on
```

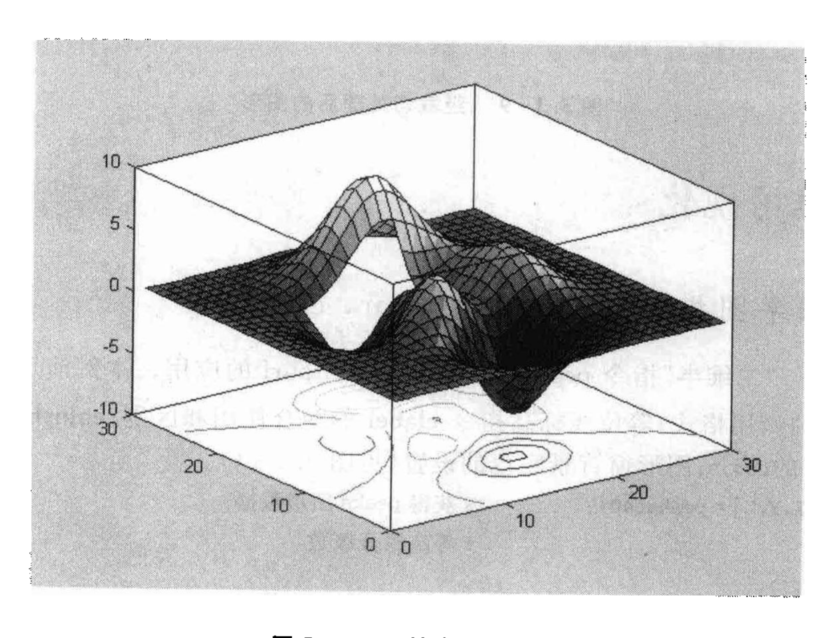


图 5.3-8　镂方孔的曲面

3. 裁　切

由 NaN 处理的图形不会产生切面。如果为了看清图形而需要表现切面，那么应该把被切部分强制为零。

例【5.3-8】 表现切面(见图 5.3-9)。

```
clf,x = [ - 8:0.1:8];y = x;[X,Y] = meshgrid(x,y);ZZ = X.^2 - Y.^2;
```

```
ii = find(abs(X)>6|abs(Y)>6);          % 确定超出[-6,6]范围的格点下标
ZZ(ii) = zeros(size(ii));              % 强制为 0
surf(X,Y,ZZ),shading interp;colormap(copper)
light('position',[0,-15,1]);lighting phong
material([0.8,0.8,0.5,10,0.5])
```

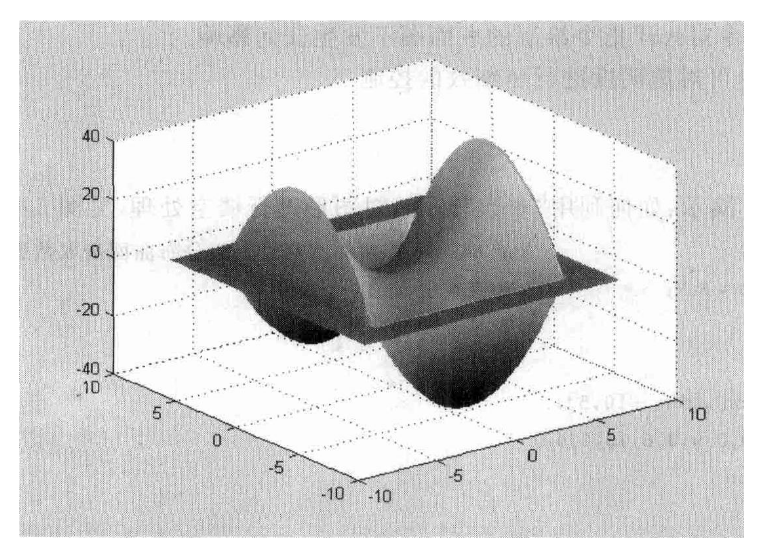


图 5.3-9 经裁切处理后的图形

5.4 高维可视化

5.4.1 二维半图指令 pcolor，contour，contourf

【例 5.4-1】 "二维半"指令 pcolor，contour，contourf 的应用。本例演示：pcolor，contour，contourf 的调用格式；等位线标高指令 clabel 的配合使用和区别；colorbar 创建色标尺，及与之配合使用的caxis；图形窗背景底色的设置（见图 5.4-1）。

```
clf;clear;[X,Y,Z] = peaks(40);              % 获得 peaks 图形数据
n = 6;                                      % 等高线分级数
subplot(1,2,1),
pcolor(X,Y,Z)                               % 伪彩图：由 X,Y 在平面上确定位置,据 Z 的大小着色
shading interp
zmax = max(max(Z));zmin = min(min(Z));      % 获取 Z 中的最大、最小值
caxis([zmin,zmax])                          % 决定色标尺的范围
colorbar                                    % 画垂直色标尺：给出颜色与等位值之间的映射关系
hold on
C = contour(X,Y,Z,n,'k:');                  % 用黑虚线画等位线,并给出标识数据
clabel(C)                                   % 把"等位值"沿等位线随机标识
hold off
```

```
subplot(1,2,2)
[C,h] = contourf(X,Y,Z,n,'k:');          % 用黑虚线画等位线并填色,又输出标识数据
clabel(C,h)                              % 沿线标识法
colormap(cool)                           % 使用 cool 色图
set(gcf,'Color','w')                     % 设置图形窗的底色为白
```

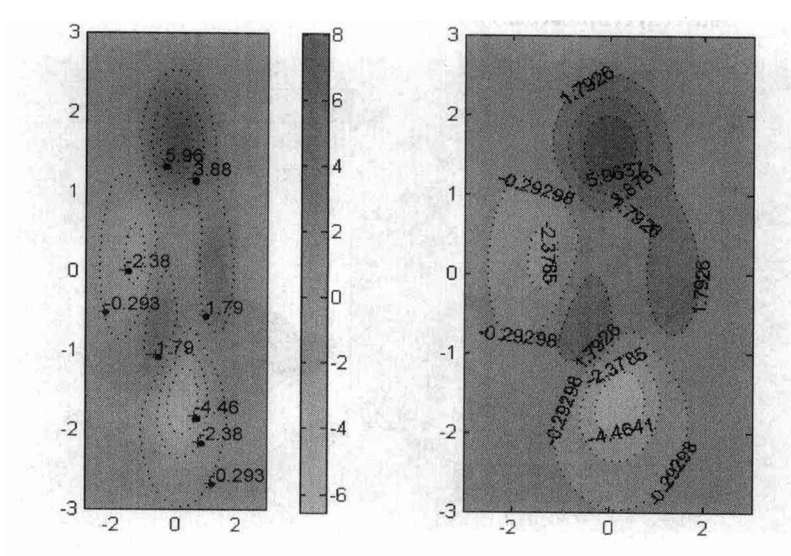


图 5.4-1　"二维半"指令的演示

5.4.2　四维表现

人对自然界的理解和思维是多维的。人的感官不仅善于接受一维、二维、三维的几何信息，而且对几何物体的运动，对颜色、声音、气味、触感等反应灵敏。从这意义上讲，MATLAB 色彩控制、动画等指令为四维或更高维表现提供了手段。

1. 准四维表现

例【5.4-2】 用颜色表现 $z=f(x,y)$ 函数的其他特征（如梯度、曲率等）。本例演示：当 surf, mesh 等指令的第四个输入量取一些特殊矩阵时，该三维图形的色彩将能表现或加强函数的某特征（如梯度和曲率）；colorbar 的使用（见图 5.4-2）。

```
clf
x = 3 * pi * ( - 1:1/15:1);y = x;[X,Y] = meshgrid(x,y);
R = sqrt(X.^2 + Y.^2) + eps;Z = sin(R)./R;
[dzdx,dzdy] = gradient(Z);
dzdr = sqrt(dzdx.^2 + dzdy.^2);          % 计算对 r 的全导数                    <4>
dz2 = del2(Z);                           % 计算曲率                             <5>
subplot(1,2,1),surf(X,Y,Z,abs(dzdr))     % 采用全导数数据给曲面着色
shading faceted;
colorbar('SouthOutside')                 % 把色标尺设置在坐标轴框的下方
brighten(0.6);                           % 增加曲面亮度
```

```
colormap hsv
title('No. 1      surf(X,Y,Z,abs(dzdr))')
subplot(1,2,2);surf(X,Y,Z,abs(dz2))      % 采用曲率数据给曲面着色
shading faceted
colorbar('NorthOutside')                 % 把色标尺设置在坐标轴框的上方
title('No. 2      surf(X,Y,Z,abs(dz2))')
```

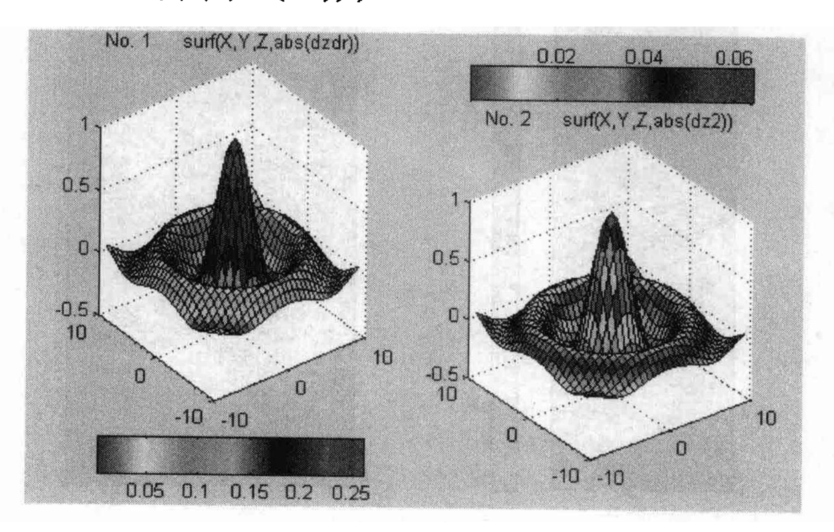


图 5.4-2　色彩分别表现函数的径向导数和曲率特征

说明

- 本例左图是根据表面关于 r 的全导数大小着色的，计算全导数的指令是第<4>条。右图是根据表面曲率大小着色的，计算曲率由指令<5>实现。
- 在有的 windows 平台上，MATLAB7 的 colorbar 可能产生不正常的表现。此时，可以尝试运行 colorbar('v6')。

2. 切片图

例 5.4-2 所介绍的仅是用色彩表现三维函数的某些特征，还不是真正意义上的四维表现。下面两个指令能真正实现四维描述。

[X,Y,Z]=meshgrid(x,y,z)	由采样向量产生三维自变量"格点"数组
slice(X,Y,Z,V,sx,sy,sz)	三元函数切片图

说明

- x, y, z 是各自变量的采样分度向量。它们的长度可以不同，比如分别是 n,m,p。而 $\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z}$ 是维数为($m\times n\times p$)的自变量"格点"数组。(注意：维数的次序。)
- $\boldsymbol{V}$ 是与 $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$ 同维的函数值数组。
- sx, sy, sz 是决定切片位置的数值向量。假如取"空阵"，就表示不取切片。

例【5.4-3】 图形表现 $v=xe^{-x^2-y^2-z^2}$。本例演示：用色彩表现三元函数的方法；slice 调用格式(见图 5.4-3)。

```
clf
```

```
[x,y,z] = meshgrid(-2:.2:2,-2:.25:2,-2:.16:2); % 三维格点阵
v = x.*exp(-x.^2-y.^2-z.^2);
xs = [-0.7,0.7]; ys = 0; zs = 0;              % 确定切片位置
slice(x,y,z,v,xs,ys,zs)                       % 产生切片图
colorbar
shading interp
colormap hsv
xlabel('x'),ylabel('y'),zlabel('z')
title('The color-to-v(x,y,z) mapping')
view([-22,39])
alpha(0.3)
```

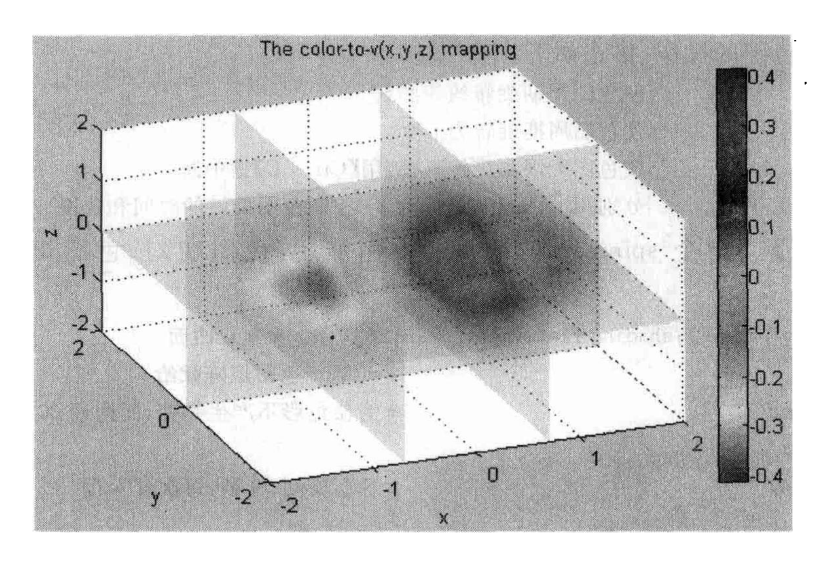


图 5.4-3　切片图

5.4.3　动态图形

在 MATLAB 的“上层”图形指令中的彗星轨线指令、色图变幻指令、影片动画指令，能很方便地使图形及色彩产生动态变化效果。

由于在 Notebook 和硬拷贝下，这种动态变化效果都无法表现，因此本节所有例题都不提供图形，而只给出有关指令。当读者在 MATLAB 指令窗中运作这些指令后，便可在图形窗中看到相应的动态图形。

1. 彗星状轨迹图

comet(x,y,p)	二维彗星轨线
comet3(x,y,z,p)	三维彗星轨线

说明

- 彗星轨线指令能动态地展示质点的运动轨迹。
- *p* 是决定彗星长度的参量。默认值为 0.1，此时二维图形中彗长 p * length(y)；三维

图形中，彗长为 p * length(z) 。

例【5.4-4】 简单二维彗星示例。（请读者自己在指令窗中运行以下指令。）

```
shg;n=2;t=n*pi*(0:0.000005:1);x=sin(t);y=cos(t);
plot(x,y,'g');axis square
hold on
comet(x,y,0.0001)
hold off
```

2. 色图的变幻

MATLAB 为颜色的动态变化提供了一个指令 spinmap。它的功能是使当前图形的色图做循环变化，以产生动画效果。与前面的动态轨迹线不同，该指令不涉及图形对象特性的操作，而只限于对色图的操作，格式如下。

spinmap	使色图周期旋转约 3 秒钟
spinmap(t)	使色图周期旋转为 *t* 秒
spinmap(inf)	使色图无限制旋转下去，用【Ctrl+C】键中断
spinmap(t,inc)	分别用 t ，inc(默认值为 2) 控制色图旋转的时间和快慢

例【5.4-5】 指令 spinmap 的应用。本例演示：色图变幻；色图矩阵的可操作性（见图 5.4-4）。

```
ezsurf('x*y','circ');shading flat;view([-18,28])     % 画双曲面
C=summer;                                             % summer 色图矩阵赋给 C
CC=[C;flipud( C )];                                   % 为使色彩不产生突变，而构造 CC
colormap(CC)
spinmap(30,4)                                         % 色彩变幻 30 秒，每次变 4 行
```

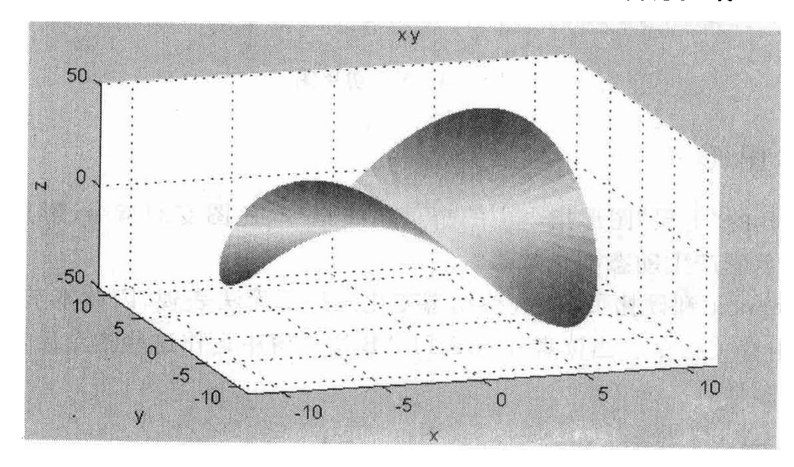


图 5.4-4　用于色图变幻演示的图形

说明

- 本例指令必须在指令窗中运行。
- 在运行本例指令后，色图矩阵的行数会成倍增加。

3. 影片动画

MATLAB 支持影片动画(movie):先把一组二维或三维图形储存起来,然后再把这组图形回放。由于人的视觉有短暂滞留,于是产生动画效果。这图形变化构成了人们观察空间的一个独立"维"。

M(i)=getframe	对当前图形拍照后产生的数据向量依次存放于画面构架数组中
movie(M,k)	以不超过每秒 12 帧的速度把 *M* 中的画面播放 *k* 次

说明

- 新版中,由 M(i)=getframe 所产生的 M(i)是一个构架数组。它由两个域:M(i). cdata 和 M(i). colormap。
- 动画的几个典型产生方法:(1) 改变某参数,获得一组画面。如驻波、行波的产生。(2) 对产生的某三维图形,改变观察角,获得一组画面。(3) 对产生的某三维图形,运用**rotate** 旋转指令,获得一组画面。参见例 5.4-6。

例【5.4-6】 三维图形的影片动画。(因书为印刷品,无法表现本例。请读者自己在指令窗中运行以下指令,观察结果。)

(1) 影片动画制作

```
clf
x = 3 * pi * ( - 1:0.05:1);y = x;[X,Y] = meshgrid(x,y);
R = sqrt(X.^2 + Y.^2) + eps;   Z = sin(R)./R;
h = surf(X,Y,Z);colormap(jet);axis off
n = 12;mmm = moviein(n);                    % 预设画面矩阵。新版完全可以取消此指令 。
for i = 1:n
    rotate(h,[0 0 1],25);                   % 使图形绕 z 轴旋转 25 度/每次
    mmm(:,i) = getframe;                    % 捕获画面。新版改为 mmm(i) = getframe 。
end
close
```

(2) 影片动画的播放

```
shg,axis off
movie(mmm,5,10)                             % 以每秒 10 帧速度,重复播放 5 次。
```

4. 实时动画

所谓实时动画通常是指:保持图形窗中绝大部分的像素颜色不变,而只更新部分像素的颜色构成运动图象。这种动画适用于每次变化较少、图形精度要求不很高的场合。制作实时动画的基本步骤:

- 画制活动对象的初始位置或/和运动轨迹图形;
- 计算活动对象的新位置;
- 在新位置上将活动对象显示出来,并将其擦除属性'EraseMode'设置为异或模式'**xor** ';
- 依靠指令 drawnow,擦除原位置上原有对象,刷新屏幕;
- 重复执行位置计算、擦除和刷新操作。

例【5.4-7】 制作红色小球沿一条带封闭路径的下旋螺线运动的实时动画（见图 5.4-5）。

（1）创建函数文件 anim_zzy1.m

编写 anim_zzy1.m 函数文件，并将其存放在 MATLAB 的 work 文件夹上。

```
function f = anim_zzy1(K,ki)
    % anim_zzy1.m          演示红色小球沿一条封闭螺线运动的实时动画
    % 仅演示实时动画的调用格式为                    anim_zzy1(K)          此时默认 ki = n/2
    % 既演示实时动画又拍摄照片的调用格式为          f = anim_zzy1(K,ki)
    % K                    红球运动的循环数(不小于 1)
    % ki                   指定拍摄照片的瞬间，取 1 到"自变量采样总点数 n"间的任意整数。
    % f                    存储拍摄的照片数据，可用 image(f.cdata)观察照片。
    % 产生封闭的运动轨线
    t1 = (0:1000)/1000 * 10 * pi;x1 = cos(t1);y1 = sin(t1);z1 = - t1;
    t2 = (0:10)/10;x2 = x1(end) * (1 - t2);y2 = y1(end) * (1 - t2);z2 = z1(end) * ones(size(x2));
    t3 = t2;z3 = (1 - t3) * z1(end);x3 = zeros(size(z3));y3 = x3;
    t4 = t2;x4 = t4;y4 = zeros(size(x4));z4 = y4;
    x = [x1 x2 x3 x4];
    n = length(x);
    if nargin<2                                          % 假如 ki 不指定，则 ki 默认为 n/2
        ki = fix(n/2);
    end
    y = [y1 y2 y3 y4];z = [z1 z2 z3 z4];
    shg
    plot3(x,y,z,'Color',[1,0.6,0.4],'LineWidth',2.5)     % 绘制轨迹曲线
    axis off
    % 定义活动对象的颜色、点形、大小、和擦除方式
    h = line('xdata',x(1),'ydata',y(1),'zdata',z(1),'Color',[1 0 0],'Marker', '.', 'MarkerSize',40,'
EraseMode','xor');
    KK = K * n;
    text( - 1, - 0.85, - 36,'倒计数')                    % 倒计数文字标注
    KK = KK - 1;
    htext = text( - 1, - 1, - 40,int2str(KK));
    % 使小球运动
    i = 2;j = 1;
    while 1                                              % 无穷循环
        set(h,'xdata',x(i),'ydata',y(i),'zdata',z(i));   % 小球位置取轨迹数据
        drawnow;                                         % 刷新屏幕               <23>
        pause(0.0005)                                    % 控制球速               <24>
        i = i + 1;
        KK = KK - 1;
        set(htext,'string',int2str(KK))                  % 动态倒计数             <27>
        if nargin = = 2 && nargout = = 1     % 仅当输入宗量为 2、输出宗量为 1 时，才拍摄照片
```

```
        if(i = = ki&&j = = 1);f = getframe(gcf);end          % 拍摄 i = ki 时的照片      <29>
    end
    if i>n
        i = 1;j = j + 1;
        if j>K;break;end
    end
end
```

(2) 在指令窗中运行以下指令，就可看到实时动画图形。

```
shg
K = 2;f = anim_zzy1(2,450);
```

(3) 显示拍摄的照片

```
image(f.cdata),axis off
```

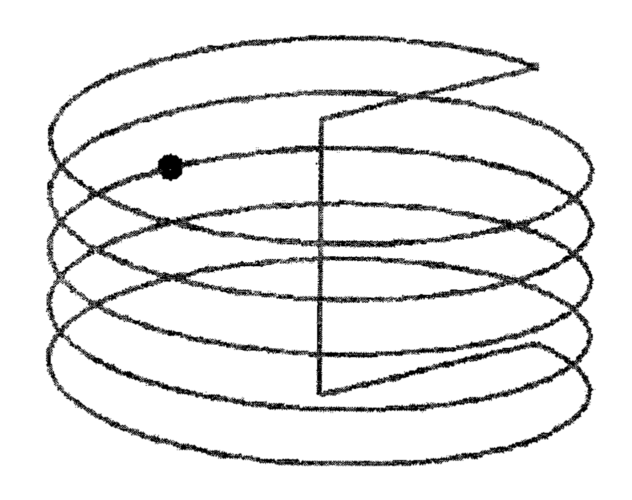


图 5.4－5　红球沿下旋螺线运动的瞬间照片

说明

- 采用 xor 擦除模式可保证螺线的色彩正确。
- 在动画演示中，红球的运动速度，受计算机的主频、一个循环中数据点数影响。本例在数据点不变情况下，为降低速度，调用了 pause 指令。
- 本例 anim_zzy1.m 程序中，由于指令〈24〉pause 的存在，屏幕一定会得到及时更新。因此，实际上指令〈23〉drawnow 在这种情况下是多余的。
- 动态倒计数由指令〈27〉实现。
- 一般说来，在实时动画中，为更新屏幕，drawnow 是必须的。
- 瞬间拍摄在指令〈29〉中实现。

5.5　图形窗功能简介

MATLAB 图形窗不仅仅是一个被动的显示窗口，而且是一个可以对图形进行编辑的交互界面。编辑操作可分为两类：一，实现“类似高层图形指令”的操作；二，实现“对图形对象属

性的编辑"操作。本文仅介绍最常用的"类似高层图形指令"的操作。至于其他功能,请参看参考文献[1]。

MATLAB 图形窗中除了 Windows 标准按键外,还有如图 5.5-1 所示的 MATLAB 特有按键。

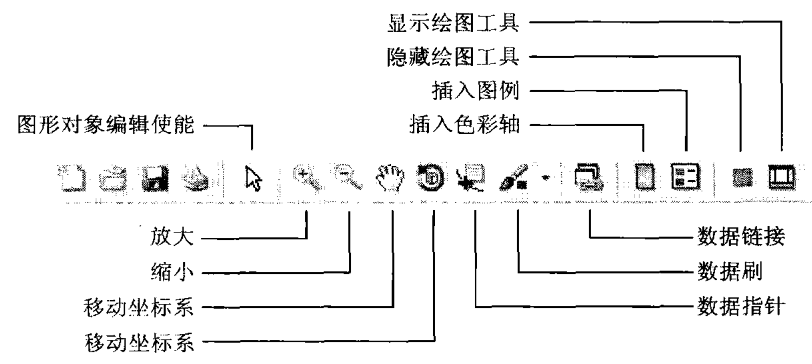


图 5.5-1　图形窗工具条专用按键

说明

- 图形对象编辑使能键:按动该键后,用鼠标双击图形对象,便进入相应的编辑状态。
- 放大键:当按下此键后,可以用鼠标左键单击并拖拉的方法,对全图或局部加以放大。若按动鼠标右键,则缩小图形。
- 缩小键:其作用与放大键相反。
- 坐标系移动转:按下此键拖动鼠标可以移动图形。
- 坐标系旋转键:一旦按下此键,光标变成带箭头的圆圈,按住左鼠标键,此时图形将随鼠标的移动而转动。注意:在旋转过程中,图形窗左下角将用方位角、俯视角数据对(az, el)实时地显示当前的观察位置。据此,用户可以再通过指令 view([az, el]),使最佳观察重现。
- 数据指针 Data Cursor 键:按下此键,在图形上点击鼠标左键可以显示"与之最临近的数据点"的数据。
- 数据刷 Data brush 键:在数据链接键按下后,再按下此键,若用鼠标在图形上选中某些区域的曲线或曲面,便引起图形对象颜色变得醒目,同时在变量编辑器中的相应数据也被用相应色彩亮化。反之亦然。
- 数据链接 Data link 键:按下此键,使数据图形与变量编辑器相链接。
- 插入色彩轴键:按下此键,在图形窗中增添色轴。
- 插入图例键:此键作用是用来加入不同的图例以区分不同参数在图形上的表示。
- 隐藏绘图工具键:该按键与"显示绘图工具栏键"是一对互逆操作键。
- 显示绘图工具键:单击该键,图形窗将由默认形式转换为编辑形式。

例【5.5-1】 利用图形窗的编辑功能,绘制图 5.5-2 所示的连续调制波形 $y=\sin(t)\sin(9t)$ 及其包络线。

(1) 运行以下指令绘出简图

```
t=(0:pi/100:pi)';                    %长度为 101 的时间采样列向量
```

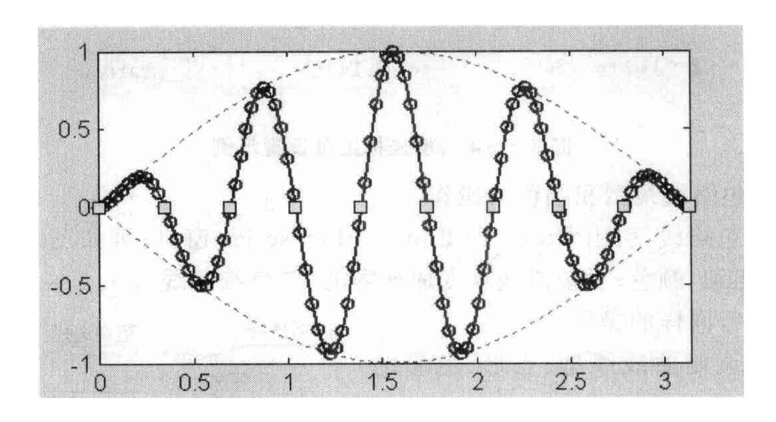


图 5.5-2　个性化的图形

```
y1 = sin(t) * [1, -1];              % 包络线函数值,是(101×2)的矩阵
y2 = sin(t). * sin(9 * t);          % 长度为 101 的调制波列向量
t3 = pi * (0:9)/9;                  % 曲线过零的时刻
y3 = zeros(size(t3));
plot(t,y1,t,y2,t3,y3)
```

(2) 使图形窗工作在编辑状态

单击“显示绘图工具栏键”,图形窗就从默认的显示模式转变为编辑模式,如图 5.5-3 所示。注意:图 5.5-1 上的中文词条是本书作者为说明本例而添加的。

(3) 进行改变坐标范围的操作

用鼠标单击图形的坐标框,在坐标框上就出现小黑方块的“图柄”(见图 5.5-3),这表明此时坐标轴处于可编辑状态;与此同时,在整个界面下方的“Property Editor—”子窗名的后注为 Axes;然后参照图 5.5-4,在 X Limits 栏中,将横坐标上限值改为 pi,按下“回车”键;于是,X Limits 栏变为图 5.5-4 所示形式。与此同时,在图 5.5-3 中,原先所绘曲线与横坐标右边界间的空白消失,即坐标轴与所绘曲线区间在横坐标方向一致。

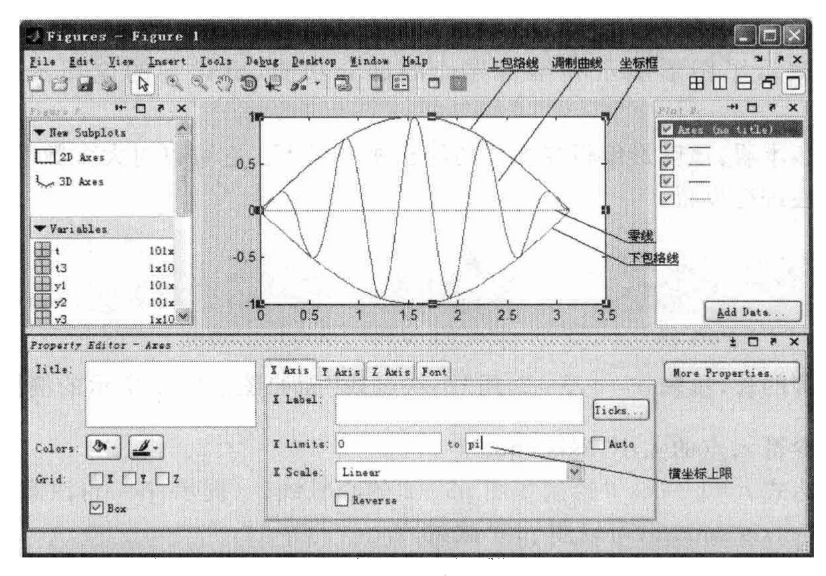


图 5.5-3　编辑工作模式下的图形窗界面

X Limits: 0 to 3.1416 Auto

图 5.5-4 横坐标上限设置示例

(4) 进行改变包络线线型和颜色的操作

用鼠标单击上包络线，引出 Property Editor—Lineseries 窗口；再如图 5.5-5 所示，在相应栏中选定线型、粗细、颜色；上包络线就变成所需的“红色细虚线”。

对下包络线进行同样的操作。

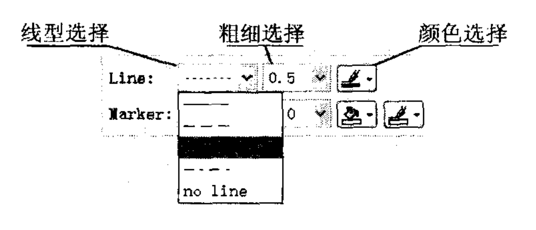


图 5.5-5 包络线的设置示例

(5) 进行改变调制曲线线型、点形、色彩的操作

用鼠标单击调制曲线，并在引出的 Property Editor—Lineseries 窗口，进行如图 5.5-6 那样的设置：Line 的粗细选 1.5，色彩为蓝；Marker 点形选空心小圆圈，大小为 6。

(6) 进行过零点的点形和色彩设置

用鼠标单击零线，并在引出的 Property Editor—Lineseries 窗口，进行如图 5.5-7 所示的设置：Line 线型为 no line；Marker 点形选择小方块，大小为 6，方块内色设置为黄，方块边线颜色选择黑。

至此，题目所要求的曲线绘制完成。

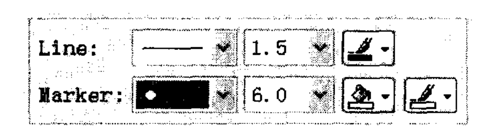


图 5.5-6 调制曲线的设置示例

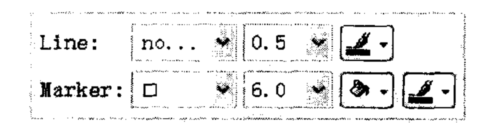


图 5.5-7 调制曲线的设置示例

说明

- 本例仅演示了图形显示和编辑界面的部分功能，更详细的应用示例请参看参考文献[1]。
- 从目前版本看，该图形编辑界面的功能已相当强大。它可以对大多数图形对象的绝大多数属性进行设置。

习题 5

1. 已知椭圆的长、短轴 $a=4$，$b=2$，用“小红点线”画如图 p5-1 所示的椭圆 $\begin{cases} x=a\cos t \\ y=b\sin t \end{cases}$。（提示：参量 t；点的大小；axis equal）
2. 根据表达式 $\rho=1-\cos\theta$ 绘制如图 p5-2 的心脏线。（提示：polar；注意 title 中特殊字符；线宽；axis square；可以用 plot 试试。）
3. A，B，C 三个城市上半年每个月的国民生产总值表 p5-1。试画出如图 p5-3 所示的

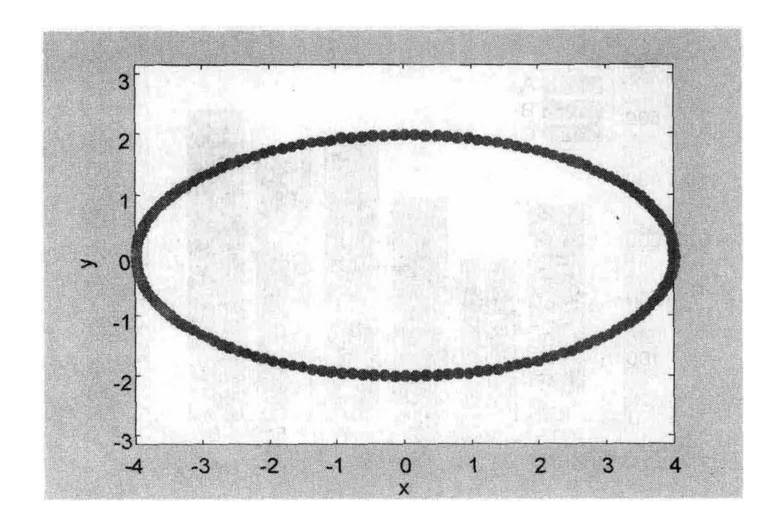


图 p5－1

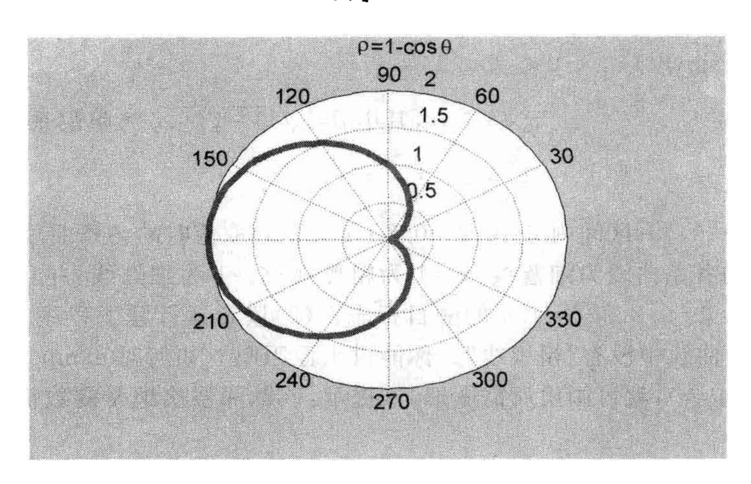


图 p5－2

三城市上半年每月生产总值的累计直方图。（提示：bar(x, Y,'style')；colormap(cool)；legend。）

表 p5－1　各城市生产总值数据(单位:亿元)

城市	1 月	2 月	3 月	4 月	5 月	6 月
A	170	120	180	200	190	220
B	120	100	110	180	170	180
C	70	50	80	100	95	120

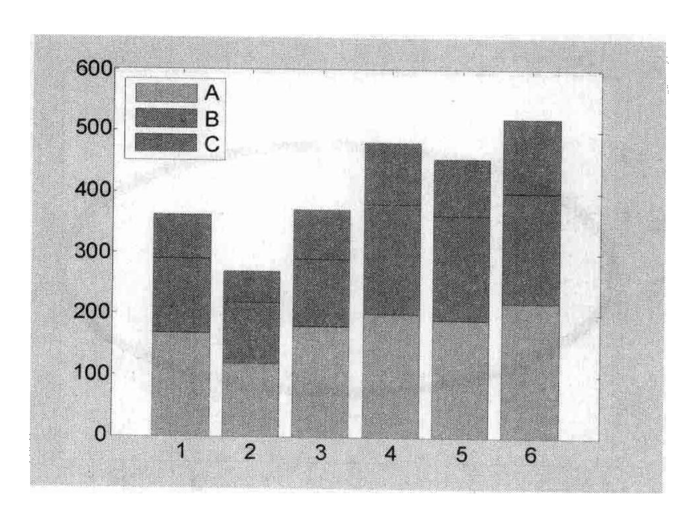


图 p5-3

4. 二阶线性系统的归一化(即令 $\omega_n=1$)冲激响应可表示为：

$$y(t)=\begin{cases}\dfrac{1}{\beta}e^{-\zeta t}\sin(\beta t) & 0\leqslant\zeta<1\\ te^{-t} & \zeta=1\\ \dfrac{1}{2\beta}\left[e^{-(\zeta-\beta)t}-e^{-(\zeta+\beta)t}\right] & \zeta>1\end{cases}$$

，其中 $\beta=\sqrt{|1-\zeta^2|}$，ζ 为阻尼系数。(1)希望在同一张图上，绘制 $t\in[0,18]$区间内 $\zeta=0.2:0.2:1.4$ 不同取值时的各条曲线(参见图 p5-4)。在此图上，$\xi\zeta<1$ 的各条曲线为细蓝线；$\zeta=1$ 为粗黑线；$\zeta>1$ 为细红线；并且对最上方及最下方的两条曲线给出 $\zeta=0.2$ 和 $\zeta=1.4$ 的醒目标志。(2)读者运行题下程序 exmp504.m，可以发现该程序画出的曲线中没有"粗黑线"。你能讲出原因吗？如何对 exmp504.m 作最少的修改(比如只改一条指令)，就可画出所需图形。(提示：该题深层次地暴露数值计算可能存在的隐患。)

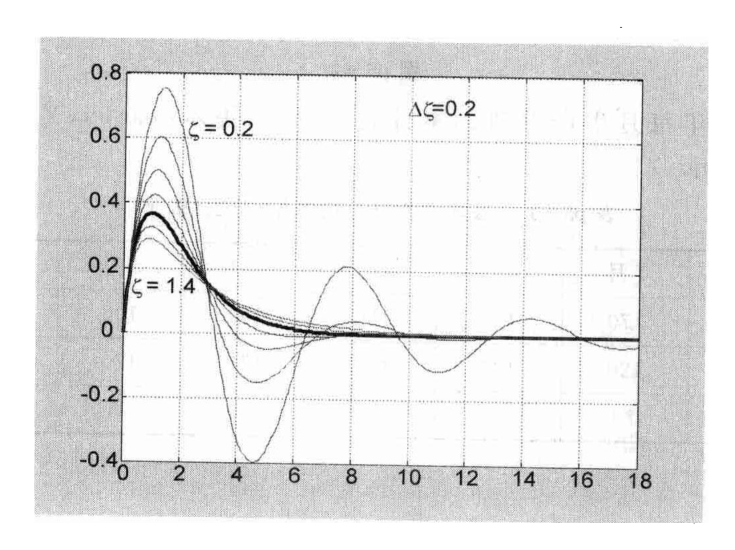


图 p5-4

```
% exmp504.m                                供第 4 道习题使用的程序
clc,clf,clear;
t = (0:0.05:18)';
N = length(t);
zeta = 0.2:0.2:1.4;
L = length(zeta);
y = zeros(N,L);
hold on
for k = 1:L
    zk = zeta(k);
    beta = sqrt(abs(1 - zk^2));
    if zk<1                                % 满足此条件,绘蓝色线
        y = 1/beta * exp( - zk * t). * sin(beta * t);
        plot(t,y,'b')
        if zk<0.4
            text(2.2,0.63,'\zeta  =  0.2')
        end
    elseif zk = = 1                        % 满足此条件,绘黑色线
        y = t. * exp( - t);
        plot(t,y,'k','LineWidth',2)
    else                                   % 其余,绘红色线
        y = (exp( - (zk - beta) * t) - exp( - (zk + beta) * t))/(2 * beta);
        plot(t,y,'r')
        if zk>1.2
            text(0.3,0.14,'\zeta  =  1.4')
        end
    end
end
text(10,0.7,'\Delta\zeta = 0.2')
axis([0,18, - 0.4,0.8])
hold off
box on
grid on
```

5. 用绿实线绘制 $x=\sin(t)$, $y=\cos(t)$, $z=t$ 的三维曲线,曲线如图 p5－5 所示。(提示:参变量;plot3;线色线粗。)
6. 采用两种不同方法绘制 $z=4xe^{-x^2-y^2}$ 在 $x,y\in[-3,3]$的如图 p5－6 的三维(透视)网格曲面。(提示:ezmesh; mesh; hidden)
7. 在 $x,y\in[-4\pi,4\pi]$区间里,根据表达式 $z=\dfrac{\sin(x+y)}{x+y}$,绘制如图 p5－7 所示的曲面。(提示:NaN 的处理)

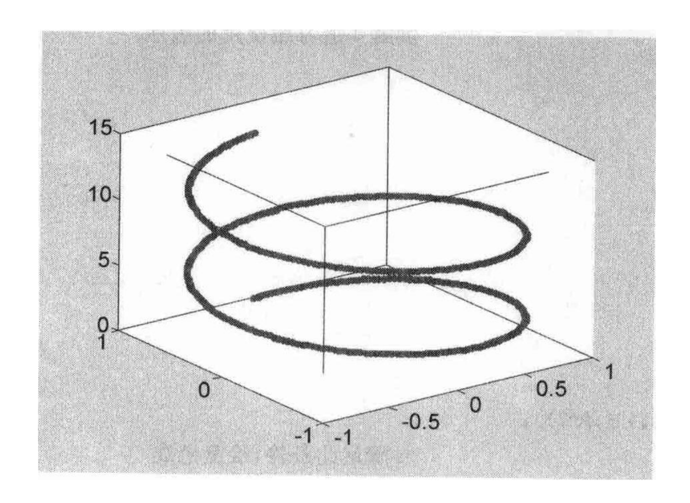


图 p5－5

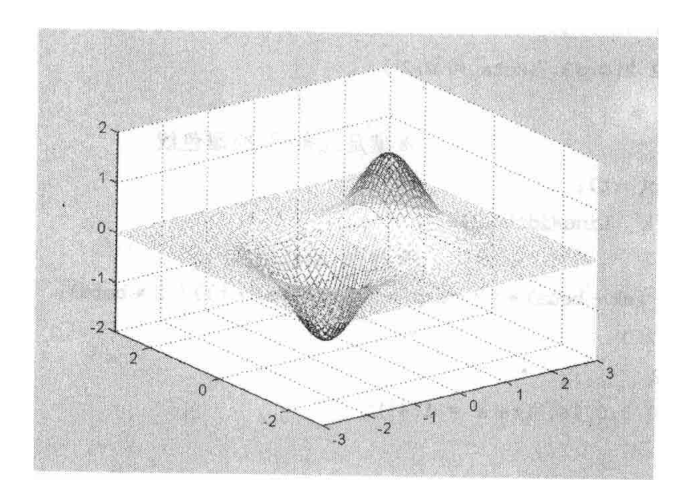


图 p5－6

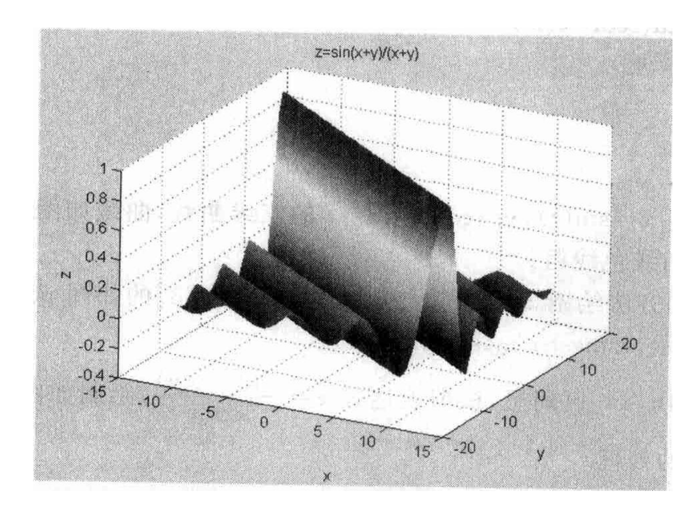


图 p5－7

8. 试用图解法回答：

(1) 方程组 $\begin{cases}\dfrac{y}{(1+x^2+y^2)}=0.1\\ \sin(x+\cos y)=0\end{cases}$ 有多少个实数解？（提示：图解法；ezplot；ginput）

(2) 求出离 $x=0, y=0$ 最近、且满足该方程组的一个近似解。

9. 制作如文件 prob509.p(在光盘的 mfiles 文件夹上)运行时那样的色图变幻(参见图 p5－8)。(提示：[jet；flipud(jet)]；colormap；spinmap)

10. 在 $[0,4\pi]$ 区间内，根据 $y(t,x)=\mathrm{e}^{-0.2x}\sin\left(\frac{\pi}{24}t-x\right)$，通过如图 p5－9 所示曲线表现"行波"。做题前，请先运行 prob510.p 文件，观察演示。(提示：采用实时动画；使用两个 line 对象；background 擦除模式；使用 pause 控制动画速度。)

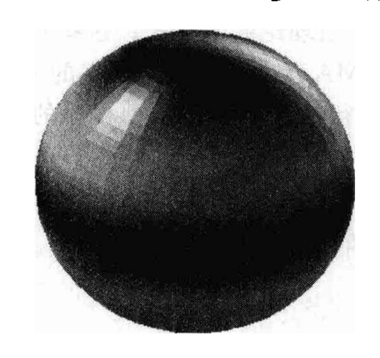


图 p5－8

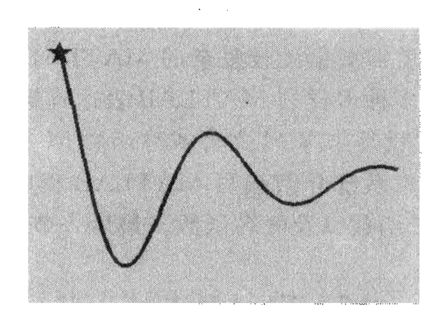


图 p5－9

11. 利用影片动画法，据函数 $f(x,t)=\sin(x)\sin(t)$ 制作如图 p5－10 所示驻波动画。在做题前，先运行 prob511.p 产生的演示动画。(提示：用 2 个 line 分别产生带图柄的线和点对象；擦除模式为 background；用 set 通过线图柄操作线位置；getframe；movie)

12. 编写使红色小球沿三叶线 $\rho=\cos(3\theta)$ 运动的程序。具体参见演示程序 prob512.p(在光盘的 mfiles 文件夹上)的运行实况。图 p5－11 显示的是该动画中的一个静止图形。(提示：用参量方程表达三叶线；用 line 绘制线对象；用 line 创建红点的图柄，擦除模式用 xor；用 set 操作红点坐标，构成动画；drawnow。)

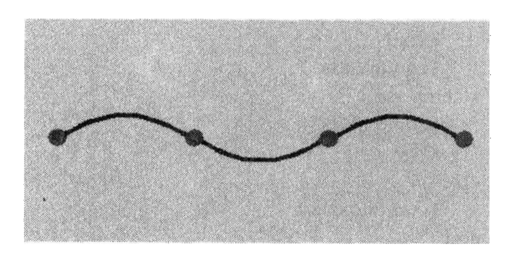


图 p5－10

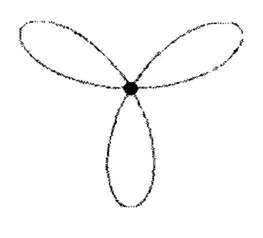


图 p5－11

第6章 M文件和函数句柄

从总体上说，单独的 MATLAB 指令及由若干指令构成的孤立表达式，只能解算或仿真简单理论数学模型所描述的问题。复杂的问题通常需要用一组理论数学算式。这些问题的解算及仿真就需要借助较复杂的 MATLAB 程序进行。这种 MATLAB 程序由较多的 MATLAB 指令和多种多样的 MATLAB 表达式组成，并循一定的执行次序运行。这种程序的扩展名为 .m。这就是脚本 M 文件或称函数 M 文件。

本章系统介绍编写 MATLAB 程序时最常用到的四种控制结构、M 函数文件的构造、主函数、子函数以及匿名函数。最后一节专门叙述函数句柄的创建、观察和调用。

6.1 MATLAB 控制流

作为一种计算机编程语言，MATLAB 提供了多种用于程序流控制的描述关键词（Keyword）。本节只介绍其中最常用的条件控制（if，switch）和循环控制（for，while，continue，break）。由于 MATLAB 的这些指令与其他语言相应指令的用法十分相似，因此本节只结合 MATLAB 给定的描述关键词，对这四种指令进行简要的说明。

6.1.1 if-else-end 条件控制

if-else-end 指令为程序流提供了一种分支控制，它最常见的使用方式见表 6.1－1。

表 6.1－1 if-else-end 分支结构的使用方式

单分支	双分支	多分支
if expr (commands) end	if expr (commands1) else (commands2) end	if expr1 (commands1) elseif expr2 (commands2) …… else (commandsk) end
当 expr 给出“逻辑 1”时，(commands)指令组才被执行	当 expr 给出“逻辑 1”时，(commands1)指令组被执行；否则，(commands2)被执行	expr1，expr2，… 中，首先给出“逻辑 1”的那个分支的指令组被执行；否则，(commandsk)被执行。该使用方法常被 switch-case 所取代

说明

- expr 是控制其下分支的条件表达式，它通常是关系、逻辑运算构成的表达式；该表达式的运算结果是“标量逻辑值 1 或 0”。expr 也可以是一般代数表达式，此时，给出的任何非零值的作用等同于“逻辑 1”。
- 在 MATLAB 中，expr 允许进行数组之间的关系、逻辑运算，因此 expr 可能给出逻辑数组。在这种情况下，只有当该逻辑数组为全 1 时，该 expr 控制的分支才执行。当 expr 给出数值数组时，只有当该数组不包含任何零元素时，该 expr 控制的分支才执行。
- 如果 expr 为空数组，MATLAB 认为条件为假（false），则该 expr 控制的分支不被执行。

例【6.1－1】 已知函数 $y=\begin{cases} x & x<-1 \\ x^3 & -1\leqslant x<1 \\ e^{-x+1} & 1\leqslant x \end{cases}$，编写能对任意一组输入 x 值求相应 y 值的程序。

（1）编写 M 函数文件

```
function y = exm060101(x)
% y = exm060101(x)     Function calculate of example 6.1 - 1

n = length(x);
for k = 1:n
    if x(k)< - 1
        y(k) = x(k);
    elseif x(k)> = 1
        y(k) = exp(1 - x(k));
    else
        y(k) = x(k)^3;
    end
end
```

（2）把文件 exm060101.m 放置在搜索路径上

（3）运行以下指令

```
x = [ - 2, - 1.2, - 0.4,0.8,1,6]
y = exm060101(x)
x =
   - 2.0000    - 1.2000    - 0.4000     0.8000     1.0000     6.0000
y =
   - 2.0000    - 1.2000    - 0.0640     0.5120     1.0000     0.0067
```

6.1.2　switch-case 控制结构

switch-case 控制结构的指令格式及其含义列于表 6.1－2。

表 6.1-2 switch-case 控制结构的使用方式

指令格式	含 义
switch expr	● expr 为根据此前给定变量进行计算的表达式
case value_1	● value_1 是给定的数值、字符串标量(或胞元数组)
(commands1)	● 若 expr 结果与 value_1(或其中的胞元元素)相等,则执行
case value_2	
(commands2)	
case value_k	● value_k 是给定的数值、字符串标量(或胞元数组)
(commandsk)	● 若 expr 结果与 value_k(或其中的胞元元素)相等,则执行
otherwise	● 该情况是以上的“并”的“补”
(commands)	● 若所有 case 都不发生,则执行该组命令
end	

说明

- 当遇到 switch 结构时,MATLAB 将表达式 expr 的值依次和各个 case 指令后面的检测值进行比较。如果比较结果为假,则取下一个检测值再来比较;而一旦比较结果为真,MATLAB 将执行相应的一组命令,然后跳出该结构。如果所有的比较结果都为假,即表达式的值和所有的检测值都不等,MATLAB 将执行 otherwise 后面的一组命令。由此可见上述结构保证了至少有一组命令会得到执行。
- switch 指令后面的表达式 expr,不管是已赋过值的变量还是变量表达式,expr 的值只能是标量数值或者标量字符串。对于标量形式的表达式,比较这样进行:表达式 = = 检测值 i。而对于字符串,MATLAB 将调用函数 strcmp 来实现比较:strcmp(表达式,检测值 i)。
- case 指令后面的检测值不仅可以为一个标量值或一个字符串,还可以为一个胞元数组。MATLAB 将把表达式 expr 的值和该胞元数组中的所有元素进行比较,如果胞元数组中某个元素和表达式的值相等,MATLAB 认为此次比较结果为真,从而执行与该检测值相应的一组命令。

例【6.1-2】 已知学生的名字和百分制分数。要求根据学生的百分制分数,分别采用“满分”、“优秀”、“良好”、“及格”和“不及格”等表示学生的学习成绩。本例演示:switch 结构的用法。

```
clear;
% 定义分数段:满分(100),优秀(90-99),良好(80-89),及格(60-79),不及格(<60)。
for k = 1:10
    a(k) = {89 + k};b(k) = {79 + k};c(k) = {69 + k};d(k) = {59 + k};
end;
c = [d,c];
% 输入学生的名字和分数
A = cell(3,5);                          % 预生成一个(3*5)的空胞元数组
A(1,:) = {'Jack','Marry','Peter','Rose','Tom'}; % 注意等号两侧括号形状不同      <7>
A(2,:) = {72,83,56,94,100};             % 注意等号两侧的括号形状不同        <8>
% 根据学生的分数,求出相应的等级。
```

```
for k = 1:5
    switch A{2,k}                      % 注意"花括号"
    case 100                           % 该 case 后的 value 是一个标量数值 100
        r = '满分';
    case a                             % a 是一个元素为数值的胞元数组{90,…,99}
        r = '优秀';
    case b                             % b 是一个元素为数值的胞元数组{80,…,89}
        r = '良好';
    case c                             % c 是一个元素为数值的胞元数组{60,…,79}
        r = '及格';
    otherwise                          % 分数低于 60 的情况
        r = '不及格';
    end
     A(3,k) = {r};
end
A
A =
    'Jack '       'Marry'       'Peter'       'Rose'       'Tom'
    [  72]     [   83]     [   56]     [   94]     [ 100]
    '及格'        '良好'        '不及格'       '优秀'        '满分'
```

说明

- 本例使用了胞元数组。胞元数组的不同胞元允许放置不同类型的数据。
- 对每个胞元赋值时,被赋的值必须放置在"花括号"内。
- 注意 a, b, c 都是胞元数组。关于胞元数组的说明,请参见附录 A.2。
- switch 工作机理说明:比如,因为 $A\{2,3\}$ 是数值 83,它与胞元数组 b 中的一个元素相等,所以程序被转入 case b 分支,使 r='良好'。

6.1.3 for 循环和 while 循环

for 循环结构和 while 循环结构的使用方式比较列于表 6.1-3 中。

表 6.1-3 循环结构的使用方式

for 循环	while 循环
for ix=array (commands) end	while expression (commands) end
● 变量 ix 为循环变量,而 for 与 end 之间的 commands 指令组为循环体 ● ix 依次取 array 中的元素;每取一个元素,就运行循环体中 commands 指令组一次,直到 ix 大于 array 的最后一个元素跳出该循环为止 ● for 循环的次数是确定的	● 当 MATLAB 碰到 while 指令时,首先检测 expression 的值,如其值为逻辑真(非 0),则执行组命令。当组命令执行完毕,继续检测表达式的值,若表达式值仍为真,循环执行组命令;而一旦表达式值为假时,结束循环 ● while 循环的次数是不确定的

例【6.1-3】 创建 Hilbert 矩阵。本例演示：for 循环的使用；给矩阵预配置内存空间，有利于提高运行速度；采用向量化编程，可大大提高运行速度。

(1) Hilbert 矩阵是一种著名“坏条件”矩阵。该矩阵的元素的表达式是$a(i,j)=\frac{1}{i+j-1}$。

(2) 下面是根据该表达式借助 for 循环生成 Hilbert 方阵的程序。

```
K=5;
A  = zeros(K,K) ;              % 给矩阵预配置内存空间。推荐使用！
for m = 1:K                    % 循环变量 m 依次取 1,2,…,K
    for n = 1:K                % 循环变量 n 依次取 1,2,…,K
        A(m,n) = 1/(m+n -1);
    end
end
format rat
A
format short g
```

```
A =
        1          1/2          1/3          1/4          1/5
      1/2          1/3          1/4          1/5          1/6
      1/3          1/4          1/5          1/6          1/7
      1/4          1/5          1/6          1/7          1/8
      1/5          1/6          1/7          1/8          1/9
```

(3) 矩阵空间预配置可提高运行速度（对高阶矩阵影响明显）

```
% 希望摒弃对“待创建矩阵不进行空间预配置”的不良习惯。
clear
tic                            % 启动秒表计时
K=1000;
for m = 1:K
    for n = 1:K
        A1(m,n) = 1/(m+n -1);
    end
end
t1=toc                         % 给出运行所用时间
```

```
t1 =
        17.434
```

```
% 提倡对“待创建矩阵进行空间预配置”。
tic
K=1000;
A2  = zeros(K,K);              % 给矩阵预配置内存空间
for m = 1:K
    for n = 1:K
        A2(m,n) = 1/(m+n -1);
```

```
    end
end
t2 = toc
t2 =
       2.1895
```

(4) 产生 Hilbert 矩阵的向量化编程将大大提高运行速度

```
% 应尽量采用向量化编程,但须对 MATLAB 函数有较高的认知。
tic
N = 1000;
n = repmat(1:N,N,1);
m = n';
A3 = 1./(n + m - 1);
t3 = toc
t3 =
      0.13909
```

说明

- Hilbert 矩阵当阶数较高时,将严重“病态”。其表现之一是:矩阵的条件数(用 cond 计算)很高。
- 高阶 Hilbert 矩阵求逆需要非常谨慎。假如 $\boldsymbol{A}$ 是 Hilbert 矩阵,那么 $\boldsymbol{B}$=inv($\boldsymbol{A}$)所求得的所谓逆矩阵 $\boldsymbol{B}$ 是很不可信的,$\boldsymbol{A}$ 逆阵的正确获取应是 $\boldsymbol{C}$=invhilb(length($\boldsymbol{A}$))。有兴趣的读者可以用 10 阶以上的 Hilbert 矩阵,进行验证性练习。
- 本例中 t1、t2、t3 的值受电脑配置、matlab 版本、该程序是否首次运行等因素影响,会有所变化。

例【6.1-4】 编写计算 $S=\sum_{n=1}^{N}\frac{1}{\sum_{k=1}^{n}k}$,其中 $N=\arg\min\left\{\frac{1}{\sum_{k=1}^{N}k}\leqslant\varepsilon\right\}$,$\varepsilon$ 是预先给定的控制精度。本例演示:while 循环的一种基本用法。

(1) 编写 M 函数文件 exm060104.m

```
function [S,N] = exm060104(epsilon)
% [S,N] = exm060104(epsilon)
%            Calculate the sum of a special series S = 1 + 1/(1 + 2) + … + 1/(1 + 2 + … + N)
% S          Sum of a special series
% N          The minimum among all numbers to have 1/sum(1:N)<epsilon
% epsilon    Given accuracy

k = 0;
s = 0;
d = inf;
S = 0;
while d>epsilon
    k = k + 1;
```

```
    s = s + k;                  % 计算 sum(1:k)
    d = 1/s;
    S = S + d;
end
N = k;
```

(2) 把文件 exm060104.m 保存在处于 MATLAB 搜索路径上的文件夹中。

(3) 运行以下指令(控制精度假设取 0.0001)

```
[S,N] = exm060104(0.0001)
S =
    1.9859
N =
   141
```

说明

本题可采用符号法加以检验。下列的 SINF,SN 分别给出"无穷项和",及"N 项和"。

```
syms k n N
SINF = limit(symsum(1/symsum(k,1,n),n,1,N),N,inf)
SINF =
2

N = 141;
SN = vpa(symsum(1/symsum(k,1,n),n,1,N))
SN =
1.9859154929577464788732394366197
```

例【6.1-5】 创建 n 阶魔方矩阵,限定条件是 n 为能被 4 整除的偶数。本例演示:while 循环与 break 的配合使用;input 指令的使用;魔方矩阵的性质和历史渊源。

(1) 所谓魔方矩阵(magic matrix),是指那矩阵由 $1\sim n^2$ 的正整数按照一定规则排列而成,并且每列、每行、每条对角线元素的和都等于$\frac{n(n^2+1)}{2}$。就生成规则而言,魔方矩阵可分成三类:一,n 为奇数;二,n 为不能被 4 整除的偶数;三,n 为能被 4 整除的偶数。

(2) 下面是当 n 为能被 4 整除的偶数时,生成 n 阶魔方矩阵的 M 脚本文件程序。

```
% exm060105.m          生成一类魔方矩阵,该魔方矩阵的阶 n 为能被 4 整除的偶数
% A                    为魔方矩阵
% n                    魔方矩阵的阶数
clear
clc
while 1                                                              % <6>
    n = input('请输入一个能被 4 整除的正整数! n =   ');
    if mod(n,4) = = 0                                                % <8>
        break                                                        % <9>
    end                                                              % <10>
end                                                                  % <11>
```

```
G = logical(eye(4,4) + rot90(eye(4,4)));  % 4 阶双对角全 1 逻辑阵
m = n/4;
K = repmat(G,m,m);           % 需要进行"补运算"的元素位置阵
N = n^2;
A = reshape(1:N,n,n);
A(K) = N - A(K) + 1          % 对选定元素关于(n^2 + 1)进行求补运算
```

（3）把 exm060105.m 文件放置在 MATLAB 的搜索路径上

（4）exm060105.m 的运行

在 MATLAB 指令窗中运行 exm060105.m 文件；指令窗中会出现请求用户输入矩阵的阶数的提示；若用户输入的正整数不能被 4 整除，那么请求将不断被重复。

假定输入正整数为 12，即 $n=12$，那么 exm060105.m 将创建出一个 12 阶的魔方矩阵。下面的检验是在这个假设下进行的。

（5）检查所生成矩阵的特性

魔方矩阵各列、各行、正对角线以及反对角线的元素和应等于 $\frac{n(n^2+1)}{2}$，该值被称为"标称和"。

```
s0 = round(n * (n * n + 1)/2);     % "标称和"
disp([int2str(n),' 阶魔方矩阵的标称和是',int2str(s0)])
Ns0 = round(2 * (n + 1));          % 对 n 矩阵而言，须检查 Ns0 组不同元素之和是否等于"标称和"
B = A';
SC = sum(A);                       % 各列"实际和"
SR = sum(B);                       % 各行"实际和"
Sd = sum(diag(A));                 % (正)对角"实际和"
Sdi = sum(diag(B));                % 反对角"实际和"
LS = [SC,SR,Sd,Sdi] = = s0;        % 检查各列、行、正反对角线的"实际和"是否等于"标称和"
NS = round(sum(LS));               % "实际和"等于"标称和"的元素组数
if NS = = Ns0
    disp('经验证，A 是魔方矩阵。')
else
    disp('经验证，A 不是魔方矩阵。')
end
12  阶魔方矩阵的标称和是    870
经验证，A 是魔方矩阵。
```

说明

- 在 exm060105.m 文件中，由第〈6〉到〈11〉条指令构成的 while 环，是为防止用户错误输入矩阵阶数而设计的。当 exm060105.m 运行时，指令窗里就会出现"请输入一个能被 4 整除的正整数！ n = "的提示。只有当输入数字是 4 的整数倍时，才可能跳出 while 环。
- 构成跳出 while 环条件的第〈8〉～〈10〉条指令可以用以下指令代替，即

```
if mod(n,4)~ = 0
    continue
```

```
end
break
```

● 关于魔方矩阵的说明：

(A) 在 MATLAB 帮助浏览器(help navigator)Contents 的 MATLAB\Getting Started\Matrices and Array\Matrices and Magic Squares 中有一幅文艺复兴时期德国画家、业余数学家 Albrecht Dürer 创作的版画 Melencolia I(忧郁人)。该版画中绘制着一个 4 阶魔方(magic square)。

(B) 魔方具有许多迷人的数学特性，至今仍是组合数学的一个研究课题。在 MATLAB 帮助文件的算例中经常用到魔方矩阵。

(C) 考证表明：魔方源于古代中国，时称“纵横图”，伴有浓重的神秘色彩。部分学者认为，“纵横图”始于《洛书》。[二九四，七五三，六一八]是最早文字记载的 3 阶魔方矩阵(见图 6.1-1)，称“九宫图”。它见著于公元前 1 世纪的《大戴礼记》“明堂篇”。公元 1275 年宋朝数学家杨辉著的《续古摘奇算法》中，有关于“纵横图”的专门研究。“纵横图”经由东南亚、印度、阿拉伯向西方传播。公元 15 世纪，“纵横图”再从土耳其的君士坦丁堡(现伊斯坦布尔)传入欧洲。

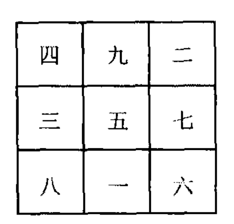


图 6.1-1 九宫图

6.1.4 控制程序流的其他常用指令

控制程序流的其他常用指令及其使用说明列于表 6.1-4 中。

表 6.1-4 控制程序流的其他常用指令

指令及使用格式	使用说明
v=input('message') v=input('message','s')	该指令执行时，“控制权”交给键盘；待输入结束，按下 Enter 键，“控制权”交还 MATLAB。message 是提示用的字符串 第一种格式用于键入数值、字符串、胞元数组等数据(见例 6.1-5)； 第二种格式，不管键入什么，总以字符串形式赋给变量 v
keyboard	遇到 keyboard 时，将“控制权”交给键盘，用户可以从键盘输入各种 MATLAB 指令。仅当用户输入 return 指令后，“控制权”才交还给程序。它与 input 的区别是：它允许输入任意多个 MATLAB 指令，而 input 只能输入赋给变量的“值”
break	break 指令，或导致包含该指令的 while，for 循环终止，或在 if-end，switch-case，try-catch 中导致中断，参见例 6.1-5
continue	跳过位于它之后的循环体中其他指令，而执行循环的下一个迭代，见例 6.1-5
pause pause(n)	第一种格式使程序暂停执行，等待用户按任意键继续； 第二种格式使程序暂停 n 秒后，再继续执行
return	结束 return 指令所在函数的执行，而把控制转至主调函数或者指令窗，否则，只有待整个被调函数执行完后，才会转出

6.2 脚本文件和函数文件

6.2.1 M脚本文件

对于一些比较简单的问题，从指令窗中直接输入指令进行计算是十分轻松简单的事。但随指令数的增加，或随控制流复杂度的增加，或重复计算要求的提出，直接从指令窗进行计算就显得烦琐。而此时脚本文件最为适宜。"脚本"本身反映这样一个事实：MATLAB只是按文件所写的指令执行。关于M脚本文件的编写，请参见1.8节。

这种文件的构成比较简单，主要有以下特点：

- 它只是一串按用户意图排列而成的(包括控制流向指令在内的)MATLAB指令集合。
- 脚本文件运行后，产生的所有变量都驻留在MATLAB基本工作空间(base workspace)中。只要用户不使用clear指令加以清除，或MATLAB指令窗不关闭，这些变量将一直保存在基本工作空间中。基本空间随MATLAB的启动而产生；只有当关闭MATLAB时，该基本空间才被删除。

6.2.2 M函数文件

与脚本文件不同，函数文件(function file)犹如一个"黑箱"，从外界只看到传给它的输入量和输出的计算结果，而内部运作是藏而不见的。M函数有以下特点：

- 从形式上看，与脚本文件不同，函数文件的第一行总是以function引导的函数申明行(function declaration line)。该行还罗列出函数与外界交换数据的全部标称输入/输出量。输入/输出量的数目并没有限制，既可以完全没有输入/输出量，也可以有任意数目的输入/输出量。
- MATLAB允许使用比标称数目少的输入/输出量，实现对函数的调用。
- 从运行上看，与脚本文件运行不同，每当函数文件运行，MATLAB就会专门为它开辟一个临时工作空间。该空间称之为函数工作空间(function workspace)。所有中间变量都存放在函数工作空间中。当执行完文件最后一条指令后，或当遇到return指令，就结束该函数文件的运行，同时该临时函数空间及其所有的中间变量就立即被清除。
- 函数空间随具体M函数文件的被调用而产生，随调用结束而删除。函数空间相对基本空间是独立的、临时的。在MATLAB整个运行期间，可以产生任意多个临时函数空间。
- 假如在函数文件中，发生对某脚本文件的调用，那么该脚本文件运行产生的所有变量都存放于那个函数空间之中，而不是存放在基本空间之中。

6.2.3 局部变量和全局变量

(1) 局部变量

存在于函数空间内部的中间变量，产生于该函数的运行过程中，其影响范围也仅限于该函数本身。正由于这种空间、时间上的局部性，中间变量被称为局部(local)变量。

(2) 全局变量

通过 global 指令,MATLAB 允许几个不同的函数空间以及基本工作空间共享同一个变量。这种被共享的变量称为全局(global)变量。每个希望共享全局变量的函数或 MATLAB 基本工作空间必须逐个用 global 对具体变量加以专门定义。没采用 global 定义的函数或基本工作空间,将无权享用全局变量。

如果某个函数的运作使全局变量的内容发生了变化,那么其他函数空间以及基本工作空间中的同名变量也就随之变化。

除非与全局变量联系的所有工作空间都被删除,全局变量依然存在。

说明

- 对全局变量的定义必须在该变量被使用之前进行。建议把全局变量的定义放在函数体的首行位置。
- 虽然 MATLAB 对全局变量的名字并没有任何特别的限制,但是为了提高 M 文件的可读性,建议选用大写字符命名全局变量。
- 由于全局变量损害函数的封装性,因此不提倡使用全局变量。

6.2.4 M 函数文件的一般结构

由于从结构上看,脚本文件只是比函数文件少一个“函数申明行”,所以只需描绘清楚函数文件的结构,脚本文件的结构也就无须多费笔墨了。

典型 M 函数文件的结构如下。

- 函数申明行(function declaration line):位于函数文件的首行;以 MATLAB 关键字 function 开头;函数名以及函数的输入/输出量名都在这一行被定义。
- H1 行(the first help text line):紧随函数申明行之后以%开头的第一注释行;按 MATLAB 自身文件的规则,H1 行包含大写体的函数文件名和运用关键词简要描述的函数功能。该 H1 行供 lookfor 关键词查询和 help 在线帮助使用。
- 在线帮助文本(help text)区:H1 行及其之后的连续的以%开头的所有注释行构成整个在线帮助文本。它通常包括:函数输入/输出宗量的含义;调用格式说明。
- 编写和修改记录:其几何位置与在线帮助文本区相隔一个空行(不用%符开头)。该区域文本内容也都以%开头;标志编写及修改该 M 文件的作者和日期、版本记录。它用作软件档案管理。
- 函数体(function body):为清晰起见,与前面的注释以空行相隔。这部分内容由实现该 M 函数文件功能的 MATLAB 指令组成。它接受输入量,进行程序流控制,创建输出量。其中为阅读、理解方便,也配置适当的空行和注释。
- 若仅从运算角度看,唯“函数申明行”和“函数体”两部分是构成 M 函数文件所必不可少的。

例【6.2-1】 编写一个 M 函数文件。它具有以下功能:(A) 根据指定的半径,画出蓝色圆周线;(B) 可以通过输入字符串,改变圆周线的颜色、线型;(C) 假若需要输出圆面积,则绘出圆。本例演示:M 函数文件的典型结构;指令 nargin, nargout 的使用和函数输入/输出量数目的柔性可变;switch-case 控制结构的应用示例;if-elseif-else 的应用示例;error 的使用。

（1）编写函数 M 文件 exm060201. m

```
function  [S,L] = exm060201(N,R,str)
% exm060201.m     The area and perimeter of a regular polygon (正多边形的面积和周长)
%               N              The number of sides
%               R              The circumradius
%               str            A line specification to determine line type/color
%               S              The area of the regular polygon
%               L              The perimeter of the regular polygon
% exm060201                    用蓝实线画半径为 1 的圆
% exm060201(N)                 用蓝实线画外接半径为 1 的正 N 边形
% exm060201(N,R)               用蓝实线画外接半径为 R 的正 N 边形
% exm060201(N,R,str)           用 str 指定的线画外接半径为 R 的正 N 边形
% S = exm060201(...)           给出多边形面积 S ,并画相应正多边形填色图
% [S,L] = exm060201(...)       给出多边形面积 S 和周长 L,并画相应正多边形填色图

%  Zhang Zhiyong 编写于 2006 - 1 - 31

switch nargin
    case  0
        N = 100;R = 1;str = '-b';
    case  1
        R = 1;str = '-b';
    case  2
        str = '-b';
    case  3
        ;                      % 不进行任何变量操作,直接跳出 switch - case 控制结构。
    otherwise
        error('输入量太多。');
end;
t = 0:2 * pi/N:2 * pi;
x = R * sin(t);y = R * cos(t);
if nargout = = 0
   plot(x,y,str);
elseif nargout>2
   error('输出量太多。');
else
   S = N * R * R * sin(2 * pi/N)/2;    % 多边形面积
   L = 2 * N * R * sin(pi/N);          % 多边形的周长
   fill(x,y,str)
end
axis equal square
box on
shg
```

(2) 把 exm060201. m 文件保存在 MATLAB 的搜索路径上，然后在指令窗中运行以下指令。

```
[S,L] = exm060201(6,2,'-g')        % 计算外接半径为 2 的正六边形面积和周长，并绘图（见图 6.2-1）
S =
   10.3923
L =
   12.0000
```

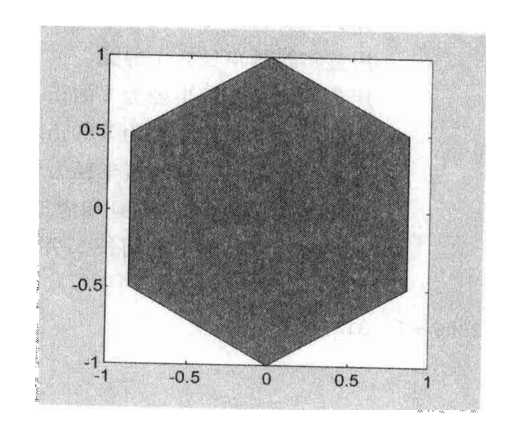


图 6.2-1 绿色正六边形

说明

- 从结构上看，M 脚本文件仅比 M 函数文件少一个"函数申明行"，其余各部分的构造和作用都相同。
- 函数定义名和保存文件名一致。两者不一致时，MATLAB 将忽视文件首行的函数定义名，而以保存文件名为准。
- 函数文件的名字必须以字母开头，后面可以是字母，下连线以及数字的任意组合，但不得超过 63 个字符。
- 读者编写 H1 行时，尽量使用英文表达，以便借助 lookfor 进行"关键词"搜索。从 MATLAB 7.0 版起，用户也可以在 H1 行中采用中文描述，现在的 lookfor 也已经支持中文搜索。
- 帮助文本区的内容可以采用英文或(和)中文书写。

6.3 MATLAB 的函数类别

MATLAB 的 M 文件分为 M 脚本文件和 M 函数文件。在 MATLAB 中，函数 Function 又被细分为主函数、子函数、嵌套函数、私用函数、匿名函数等。限于篇幅，本节只对主函数、子函数及匿名函数进行阐述。

6.3.1 主函数

主函数(primary function)的特点：

- 一般为"与保存文件同名"的那个函数；

● 在当前目录、搜索路径上，列出文件名的函数；
● 在指令窗中或其他函数中，可直接调用的函数；
● M 函数文件中，由第一个 function 引出的函数；
● 采用 help functionname 可获取函数所携带的帮助信息。

6.3.2 子函数

子函数(subfunction)的特点：
● 子函数不独立存在，只能寄生在主函数体内；
● 在函数文件中，由非第一个 function 引出的函数；
● 一个 M 函数文件可以包含多个子函数；
● 子函数只能被其所在的主函数和其他"同居"子函数调用；
● 子函数可以出现在主函数体的任何位置，其位置先后与调用次序无关；
● 在 M 函数文件中，任何指令通过"名字"对函数进行调用时，子函数的优先级仅次于内装函数；
● 同一文件的主函数、子函数的工作空间都是彼此独立的，各函数间的信息，或通过输入输出宗量传递，或通过全局变量传递；
● 采用 help functionname/subfunctionname 可获取子函数所带的帮助信息；
● 不管在什么地方，只要存在那子函数句柄，就可以直接调用子函数。

例【6.3-1】 编写一个内含子函数的 M 函数绘图文件。本例演示：内含子函数的 M 函数文件；switch-case 用法示例；函数句柄的用法示例；脱离主函数，直接利用子函数句柄的示例。

(1) 编写函数 M 文件 exm060301.m

```
function Hr = exm060301(flag )
% exm060301.m          Demo for handles of primary functions and subfunctions
%         flag         可以取字符串 'line' 或  'circle'。
%         Hr           子函数 cirline 的句柄


t = (0:50)/50 * 2 * pi;
x = sin(t);
y = cos(t);
Hr = @cirline;             % 创建子函数的句柄
feval(Hr,flag,x,y,t)
% -------------subfunction---------------------------
function cirline(wd,x,y,t)
% cirline(wd,x,y,t)  是位于 exm060301.m 函数体内的子函数
%        wd          接受字符串 'line'或'circle'
%        t           画线用的独立参变量
%        x           由 t 产生的横坐标变量
%        y           由 t 产生的纵坐标变量
switch wd
case 'line'
```

```
        plot(t, x, 'b' ,t , y, 'r', 'LineWidth', 2)
    case 'circle'
        plot(x, y, '-g', 'LineWidth', 8),
        axis square off
    otherwise
        error('输入宗量只能取"line"或 "circle" ! ')
    end
    shg
```

(2) 把 exm060301.m 文件保存在 MATLAB 的搜索路径上，然后在指令窗中运行以下指令(见图 6.3-1)。

```
HH = exm060301('circle')
HH =
    @cirline
```

(3) 直接利用创建的子函数句柄，调用子函数

```
t = 0:2 * pi/5:2 * pi;x = cos(t);y = sin(t);    % 为绘制正五边形准备数据
HH('circle',x,y,t)                              % 利用句柄绘图(见图 6.3-2)
```

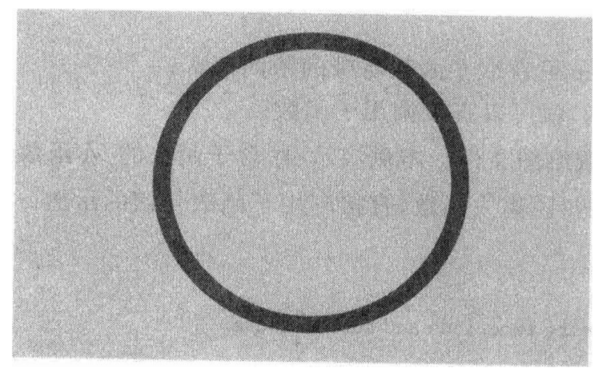


图 6.3-1　绿色圆周线

图 6.3-2　由子函数绘制的绿色正五边形

6.3.3 匿名函数

匿名函数(anonymous function)不以文件形式驻留在文件夹上；它的生成方式最简捷，可在指令窗或任何函数体内通过指令直接生成。

创建匿名函数的一般方法如下。

(1) 在指令窗或任何 M 文件中创建匿名函数

```
FH= @(arglist) expr
```

在此：FH 是所创建匿名函数的句柄；arglist 是匿名函数的输入量列表；expr 是由输入量构成的函数表达式。

(2) 匿名函数的调用格式

FH(arglist)	直接调用格式
feval(FH, arglist)	间接调用格式

说明

- arglist 中输入量的次序必须与创建该匿名函数句柄时的输入量次序相同。
- 所有借助 feval 构成的泛函指令也采用间接调用格式。关于匿名函数的演示算例参见例 4.1-6,4.1-7,4.1-8。

6.4 函数句柄

函数句柄(Function handle)是 MATLAB 的一种数据类型。它携带着“相应函数创建句柄时的路径、视野、函数名以及可能存在的重载方法”。

引入函数句柄是为了使 feval 及借助于它的泛函指令工作更可靠;使“函数调用”像“变量调用”一样灵活方便;提高函数调用速度,特别在反复调用情况下更显效率;提高软件重用性,扩大子函数和私用函数的可调用范围;迅速获得同名重载函数的位置、类型信息。

6.4.1 函数句柄的创建和观察

函数句柄并不伴随函数文件的被创建、调用而自动形成。它必须通过专门的定义才会生成。为一个函数定义句柄的方法有两种:利用@符号;利用转换函数 str2func。至于对函数句柄内涵的观察需借助专门指令 functions 实现。

例【6.4-1】 为 MATLAB 的 magic 函数创建函数句柄,并观察其内涵。

(1) 创建句柄

```
hm = @magic
hm =
    @magic
```

(2) 类型判别

```
class(hm)
isa(hm, 'function_handle')
ans =
function_handle
ans =
     1
```

(3) 借助指令 functions 观察内涵

```
CC = functions(hm)
CC =
    function: 'magic'
        type: 'simple'
        file: 'C:\MATLAB R2010a\toolbox\matlab\elmat\magic.m'
```

(4) 句柄的调用方法之一

```
M1 = hm(4)
M1 =
    16     2     3    13
     5    11    10     8
```

```
     9     7     6    12
     4    14    15     1
```

(5) 句柄的调用方法之二

```
M2 = feval(hm,4)
M2 =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

说明

- 指令 hm＝@magic 的功能，可以用 hm＝str2func('magic') 替代。
- 在创建一个函数句柄时，那函数必须处在"当前视野 Scope"内。否则，所创建的函数句柄无效。所谓当前视野是指：在当前目录下，help，lookfor，which 等指令能发挥正常作用的范围。
- 定义函数句柄时，所指定的函数名不应包括"路径信息"，也不应包括扩展名。函数名最多只能包括 63 个字符。

6.4.2 函数句柄的基本用法

在第 6.3.3 节和第 6.4.1 节中，已经简单介绍和演示了函数句柄两种调用格式的简单形式。本节将更为准确地介绍两种函数句柄调用格式的一般形式。

假设原函数 FunName 的调用格式为

[argout1, argout2, …, argoutn]＝FunName(argin1, argin2, …, arginn)

又假设该函数的句柄 Hfun 已有效地创建，那么通过函数句柄实现函数运算的调用格式是

[argout1, argout2, …, argoutn]＝Hfun(argin1, argin2, …, arginn) 直接调用格式

[argout1,argout2, …, argoutn]＝feval(Hfun,argin1, argin2, …, arginn) 间接调用格式

说明

- 不管名为 FunName 的函数文件是否在当前搜索路径上，不管它是否在当前视野内，不管它是否子函数或私用函数，只要句柄 Hfun 已经被有效创建，它代表的函数总能被正确地执行。
- 在 FunName 存在重载函数时，借助句柄 Hfun 的计算总能被"恰当地"执行。也就是说，它也会根据待计算的数据类型，从其包含的所有重载函数中选择相应的函数文件执行运算。
- 在不使用函数句柄的情况下，对 FunName 进行多次调用时，每次都要为该函数进行全面的路径搜索，而直接影响计算速度。借助 Hfun 可完全克服这种无谓的时间消耗。

例【6.4-2】 借助本书算例 6.2-1 建立的函数文件 exm060201.m，本例演示：体验"视野"概念；检验文件是否存在于当前视野之中；创建用户自建函数的函数句柄；体验已建句柄不依赖视野而正确运作的能力。

(1) 把 exm060201.m 所在的文件夹设置为当前文件夹。此时，在 MATLAB 的 Desktop 操作界面的"当前目录 Current Directory"中可以看到函数文件 exm060201.m。

(2) 在指令窗或 M-book 文件中运行以下指令，可验证：该函数存在于"当前视野"及其功能。调用函数 exm060201 画出红色等边三角形如图 6.4－1 所示。

```
[S,L] = exm060201(3,2,'-r')          % 计算外接半径为 2 的等边三角形面积和周长，并绘图。
S =
    5.1962
L =
   10.3923
```

(3) 在指令窗或 M-book 文件中运行以下指令，可有效地创建 exm060201. m 的函数句柄。

```
Hexm = @exm060201,                    % 创建函数句柄
Hexm =
    @exm060201
```

(4) 改变当前文件夹设置，把 matlab\work 设置为当前文件夹，从而改变了原先的"视野"。在新的视野中，将看不到函数文件 exm060201. m 的存在。运行以下指令，结果表明：新的当前视野中不存在 exm060201. m。

```
which('exm060201')
'exm060201' not found.
```

(5) 在新的视野中，再次直接通过函数名调用函数文件 exm060201. m，将遭失败。其原因是：exm060201. m 既不在当前文件夹上，也不在搜索路径上。

```
[S,L] = exm060201(3,2,'-r')
??? Undefined command/function 'exm060201'.
```

(6) 在新视野中，利用函数句柄，函数仍能被正常调用。其原因是：函数句柄中包含了原函数文件所在位置的信息，用函数句柄画红色等边三角形如图 6.4－2 所示。

```
[S,L] = Hexm(3,2,'-r')               % 借助句柄，可正确运作。
S =
    5.1962
L =
   10.3923
```

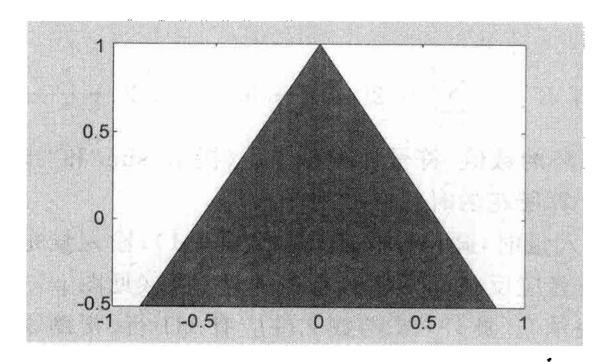


图 6.4－1　直接调用函数 exm060201 所画的红色等边三角形

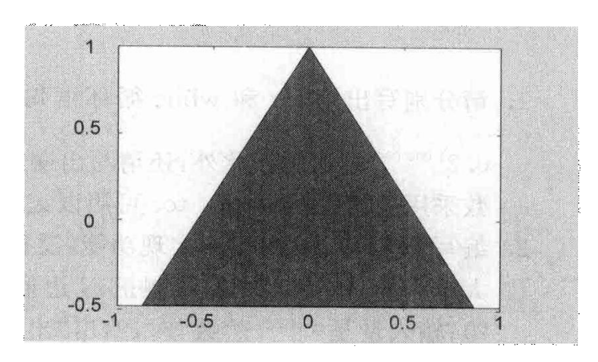


图 6.4－2　在新视野中里用函数句柄 Hexm 所画的红色等边三角形

例【6.4-3】 借助本书算例 6.2-1 建立的函数文件 exm060201.m，演示：若创建函数句柄时，原函数不在当前视野内，那么所创建的函数句柄是一个无效句柄。

(1) 运行以下指令，清除 MATLAB 工作空间中的所有变量(包括上例已创建的函数句柄 Hexm)

```
clear
```

(2) 在 work 当前目录下，运行以下指令，为 exm060201.m 进行句柄创建。注意：此时，exm060201.m 不在当前视野中，但指令运行，并没有给出任何警告；外表看起来，似乎一切正常

```
Hexm_2 = @exm060201
Hexm_2 =
    @exm060201
```

(3) 以下指令也不能判定这样所建函数句柄是否无效

```
class(Hexm_2)
isa(Hexm_2,'function_handle')
ans =
function_handle
ans =
     1
```

(4) 但在调用时，可以发现该句柄是无效的

```
[S,L] = Hexm_2(3,2,'-r')
??? Undefined command/function 'exm060201'.
```

说明

- 创建函数句柄前，一定要确认原函数在当前视野内，否则所创建的句柄是虚假的、无效的。
- 就迄今为止的 MATLAB 各版本而言，假如不是直接调用所创建函数句柄进行运作，那么 MATLAB 没有什么指令可以正确诊断所创建函数句柄的有效性。

习题 6

1. 请分别写出用 for 和 while 循环语句计算 $K=\sum_{i=0}^{1\,000\,000}0.2^i=1+0.2+0.2^2+\cdots+0.2^{1\,000\,000}$ 的程序。此外，还请写出避免循环的数值、符号计算程序。(提示：sum 和"指数采用数组"配合；tic，toc 可用以记录计算所花的时间。)
2. 编写一个 M 函数文件，实现功能：没有输入量时，画出单位圆(见图 p6-1)；输入量是大于 2 的自然数 N 时，绘制正 N 边形，图名应反映显示多边形的真实边数(见图 p6-2)；输入量是"非自然数"时，给出"出错提示"。此外，M 函数文件应有 H1 行、帮助说明和程序编写人姓名。(提示：nargin，error，int2str。)
3. 用泛函指令 fminbnd 求 $y(x)=-e^{-x}|\sin[\cos x]|$ 在 $x=0$ 附近的极小值。fminbnd 的第一个输入量要求使用匿名函数表达。(提示：注意搜索范围的选择；假如极值在边

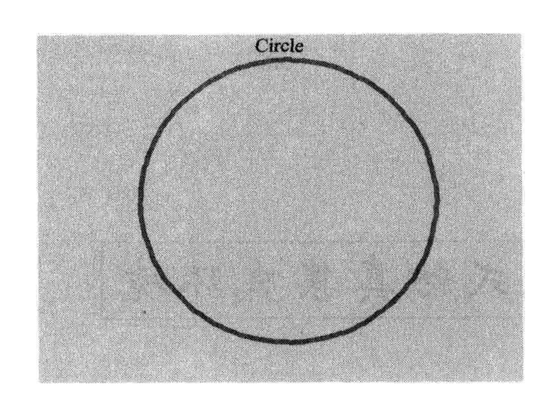


图 p6 - 1

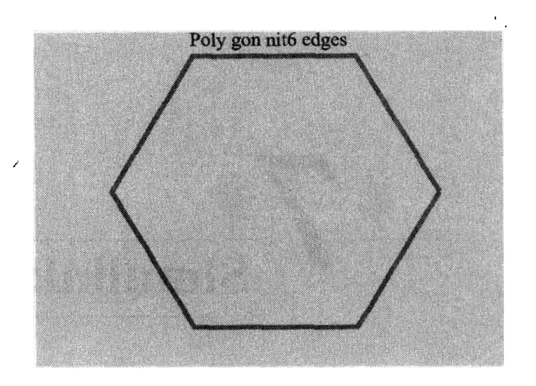


图 p6 - 2

界附近，进一步扩大搜索范围是合理的选择。）

4. 在 matlab 的 \toolbox\matlab\elmat\private 文件夹上有一个“烟圈矩阵”发生函数 smoke.m。运行指令 smoke(3,0,'double')，将生成一个 3 阶伪特征根矩阵：

$$\boldsymbol{A}=\begin{bmatrix}-0.5000+0.8660i & 1.0000 & 0\\ 0 & -0.5000-0.8660i & 1.0000\\ 1.0000 & 0 & 1.0000\end{bmatrix}$$

现在的问题是：在 MATLAB 当前目录为\work 情况下，如何利用函数句柄调用 smoke.m 函数，产生 3 阶伪特征根矩阵。请写出相应的程序或操作步骤。（提示：注意函数句柄创建的有效性；若想编写能完全自动执行的解题程序，注意使用 cd, pwd, which, 及字符串的拆分、合并操作技术。）

第7章 Simulink交互式仿真集成环境

Simulink是MATLAB最重要的组件之一，它向用户提供一个动态系统建模、仿真和综合分析的集成环境。在这环境中，用户无须书写大量的程序，而只需通过简单直观的鼠标操作，选取适当的库模块，就可构造出复杂的仿真模型。Simulink的主要优点：

- 适应面广。可构造的系统包括：线性、非线性系统；离散、连续及混合系统；单任务、多任务离散事件系统。
- 结构和流程清晰。它外表以方块图形式呈现，采用分层结构。既适于自上而下的设计流程，又适于自下而上逆程设计。
- 仿真更为精细。它提供的许多模块更接近实际，为用户摆脱理想化假设的无奈开辟了途径。
- 模型内码更容易向DSP，FPGA等硬件移植。

基于本书定位，为避免内容空泛，本节对于Simulink将不采用横断分层描述，即不对Simulink库、模块、信号线勾画标识等进行分节阐述。本节将以四个典型算例为准线，纵向描述Simulink的使用要领。

7.1 连续时间系统的建模与仿真

创建动态系统Simulink模型一般步骤如下：

- 首先写出描写系统动力学的全部方程，包括微分方程、差分方程、代数方程等。
- 打开Simulink模块库，引出Simulink工作环境。
- 为待建模型，开启空白模型窗。
- 根据理论数学模型，在Simulink模块库中选择所需的模块，并将其复制到空白模型窗。
- 假如模块结构不符合数学模型要求，那么在勾画信号线之前，必须先设置模块中影响结构的参数，如输入输出口的数目。
- 根据理论数学模型，借助鼠标勾画各模块间的信号连线，使之成为系统。
- 再根据理论数学模型，从系统角度出发，对各模块的非结构参数进行设置。
- 根据经验或采用默认方法，对仿真解算器和仿真终止时间进行设置。
- 给新建模型起适当的名字加以保存。
- 运行Simulink模块模型，并根据情况进行适当调试。
- 保存调试好的Simulink模型，以供仿真研究使用。

7.1.1 基于微分方程的 Simulink 建模

本节将从微分方程出发，以算例形式详细讲述 Simulink 模型的创建和运行。

例【7.1-1】 在图 7.1-1 所示的系统中，已知质量 $m=1$ kg，阻尼 $b=2$N. s/m，弹簧系数 $k=100$ N/m，且质量块的初始位移 $x(0)=0.05$ m，其初始速度 $x'(0)=0$ m/s，要求创建该系统的 Simulink 模型，并进行仿真运行。本例演示，据物理定理建立微分方程，并以此微分方程创建 Simulink 模型的完整步骤：微分方程的整理；模块的复制；信号线的构画；模块参数设置；示波器的调整；仿真参数设置。

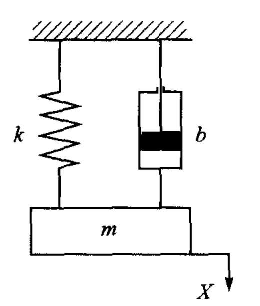


图 7.1-1 弹簧—质量—阻尼系统

(1) 建立理论数学模型

对连续动态系统而言，描述该系统动力学的微分方程或传递函数是 Simulink 建模的最原始出发点。

对于无外力作用的"弹簧—质量—阻尼"系统，据牛顿定律可写出

$$mx''+bx'+kx=0 \tag{7.1-1}$$

代入具体数值并整理，可得

$$x''=-2x'-100x \tag{7.1-2}$$

在此还要指出：为方便 Simulink 模型的构作，常把微分方程的最高阶导数项写在等号左边，而把函数及其他导数项写在等号右边。

(2) 建模的基本思路

- 采用"积分"模块，而不采用"求导"模块，描写二阶导数与一阶导数、一阶导数与函数间的关系。
- 式(7.1-2)等号右边各项的非 1 系数，借助"增益"模块实现。
- 式(7.1-2)等号右边两项的代数和运算采用"求和"模块实现。
- 为了观察位移随时间的变化，还需要显示"示波器"模块。

(3) 打开 Simulink 模块库

在 MATLAB 工作界面(Desktop)的工具条上，单击图标；或在 MATLAB 指令窗中运行 simulink，就可引出如图 7.1-2 所示的 Simulink 模块库浏览器。

(4) 开启空白(新建)模型窗

单击 Simulink 模块库浏览器工具条上的图标□；或选择下拉菜单项{File:New}，引出如图 7.1-3 所示的空白模型窗(注：窗中的模块是后一个步骤复制进去的)。

(5) 从模块库复制所需模块到空白(新建)模型窗(见图 7.1-3)

- 把连续模块子库中的积分器<Simulink\Continuous\Integrator>"拖"进空白模型窗。
- 把 Simulink\Math Operations 子库中的增益模块<Gain>"拖"进空白模型窗；再用鼠标点亮该模块，选中下拉菜单项{Format:Flip Block}，使增益模块<Gain>翻转成图 7.1-3 所示那样。
- 把 Simulink\Math Operations 子库中的求和模块<Sum>"拖"进空白模型窗。
- 再把<Simulink\Sink\Scope>示波器模块"拖"进空白模型窗。

(6) 新建模型窗中的模型再复制

库目录　图标扩展　搜索内容　搜索　搜索模式　库内容清单　搜索结果

Simulink Library Browser

File Edit View Help

Enter search term

Libraries

Library: Simulink/Commonly Used Blocks　　Search Results: (non

Simulink
Commonly Used Blo...
Continuous
Discontinuities
Discrete
Logic and Bit Operat...
Lookup Tables
Math Operations
Model Verification
Model-Wide Utilities
Ports & Subsystems
Signal Attributes
Signal Routing
Sinks
Sources
User-Defined Functi...

Bus Creator　Bus Selector
Constant　Data Type Conversion
Demux　Discrete-Time Integrator
Gain　Ground
In1　Integrator
Logical Operator　Mux
Out1　Product

Showing: Simulink/Commonly Used Blocks

所在库位置显示　所选中库　所选中模块

图 7.1-2　Simulink 模块库浏览器

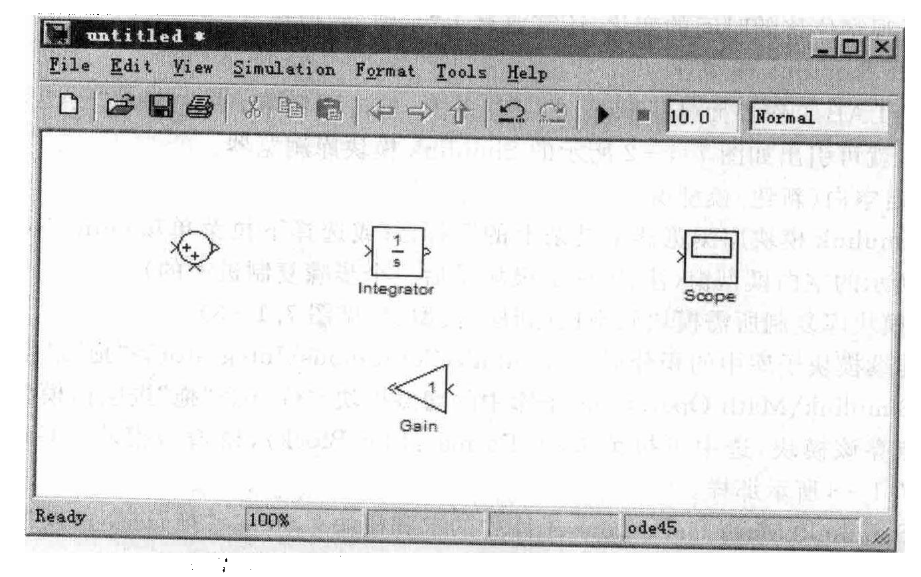


图 7.1-3　已经复制进库模块的新建模型窗

在该新建模型窗中，可以通过再复制，产生建模所需的 2 个积分模块和 2 个增益模块。具体操作方法是：按住[Ctrl]键，用鼠标"点亮并拖拉"积分模块<Integrator>到适当的位置，便完成了积分模块的再复制。而这被再复制的积分模块会自动改名为<Integrator1>，以作区分（见图 7.1－4）。

另一个所需的增益模块，可采用类似方法获得。

（7）模块间信号线的连接

信号连接线的形成方法随信号线起始端位置不同而不同，具体如下（见图 7.1－4）。

- 起始端位于模块输出口的信号线生成法。使光标靠近模块输出口；待光标变为"单线十字叉"时，按下鼠标左键；移动十字叉，拖出一根"虚连线"；光标与另一个模块输入口靠近到一定程度，单十字叉变为双十字叉；放开鼠标左键，"虚连线"便变为带箭头的信号连线。
- 起始端位于其他信号线上的信号线生成法。使光标置于已经存在的其他信号线之上；按下鼠标右键，光标变为"单线十字叉"；运动鼠标，引出"虚连线"；当光标与待连接模块输入口靠得足够近，单十字叉变为双十字叉；放开鼠标右键，"虚连线"便变为带箭头的信号连线。

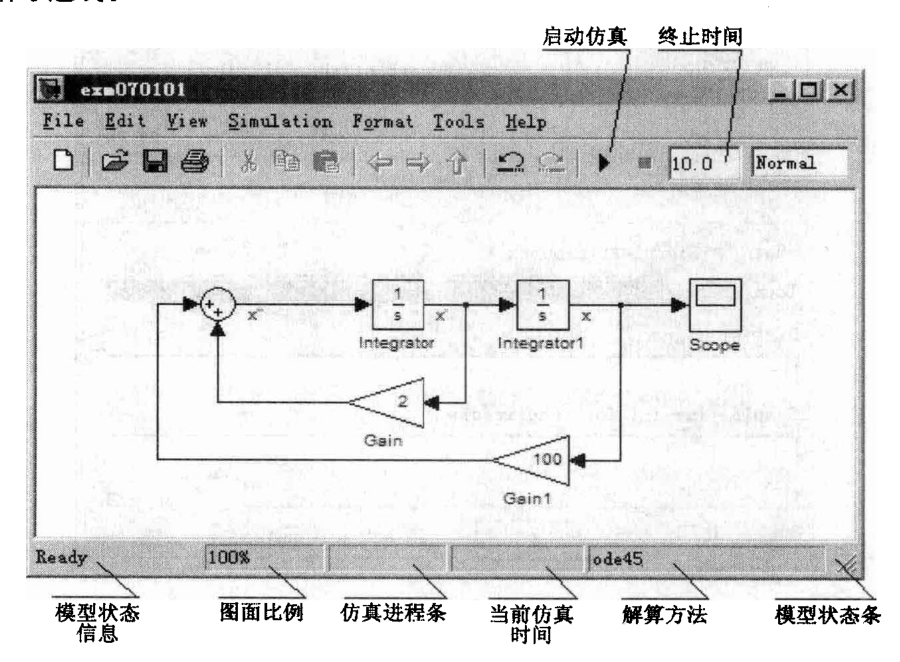


图 7.1－4　已构建完成的新模型窗

（8）根据理论数学模型设置模块参数

为使构造的 Simulink 模型与理论数学模型一致，须对模块进行如下参数设置。

- 设置增益模块<Gain>参数：双击模型窗中的增益模块<Gain>，引出如图 7.1－5 所示的参数设置窗；把 Gain 增益栏中默认数字 1 改写为所需的 2；单击 [OK]键，完成设置；此时，新建模型窗中<Gain>增益模块上会出现数字 2（参见图 7.1－4）。
- 参照以上方法，把<Gain1>增益模块的增益系数修改为 100。
- 修改求和模块输入口的代数符号：双击求和模块，引出如图 7.1－6 所示的参数设置窗；把符号列表栏（List of signs）中的默认符号（＋＋）修改成式（7.1－2）所需的代数

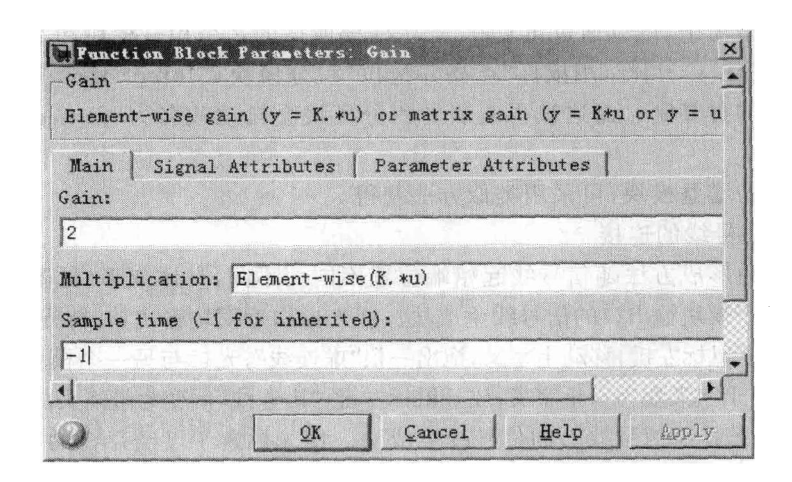


图 7.1-5 参数已经修改为 2 的＜Gain＞增益模块参数设置窗

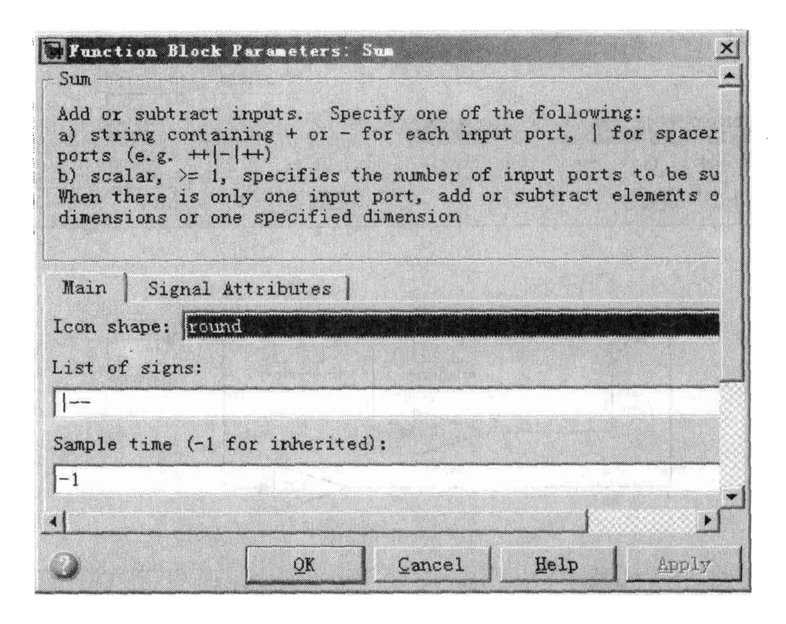


图 7.1-6 改变输入口符号的求和模块参数设置窗

符号(——)；单击 [OK]键，完成设置；此时，求和模块将呈现图 7.1-4 中的样式。

- 据初始位移 $x(0)=0.05$ m 对积分模块＜Integrator1＞的初始状态进行设置：双击积分模块＜Integrator1＞，引出如图 7.1-7 所示的参数设置窗；把初始条件 Initial condition 栏中的默认 0 初始修改为题目给定的 0.05；单击 [OK]键，关闭该窗口，完成设置。

(9) 仿真运行参数采用默认解算器“ode45”、默认“变步长”和默认仿真终止时间 10。

(10) 把新建模型保存为 exm070101.mdl。

(11) 试运行，以便发现问题加以改善。

- 双击＜Scope＞示波器模块，引出示波显示窗，并使它不与 exm070101 模型窗重叠。
- 单击 exm070101 模型窗上的▶仿真启动键，使该模型运行；在示波器上呈现的运行结果可能如图 7.1-8 所示。

Function Block Parameters: Integrator1

Integrator

Continuous-time integration of the input signal.

Parameters

External reset: none

Initial condition source: internal

Initial condition:

0.05

Limit output

Upper saturation limit:

inf

Lower saturation limit:

-inf

Show saturation port

Show state port

Absolute tolerance:

auto

Ignore limit and reset when linearizing

Enable zero-crossing detection

State Name: (e.g., 'position')

OK　Cancel　Help　Apply

图 7.1-7　实现初始位移 0.05 设置的<Integrator1>设置窗

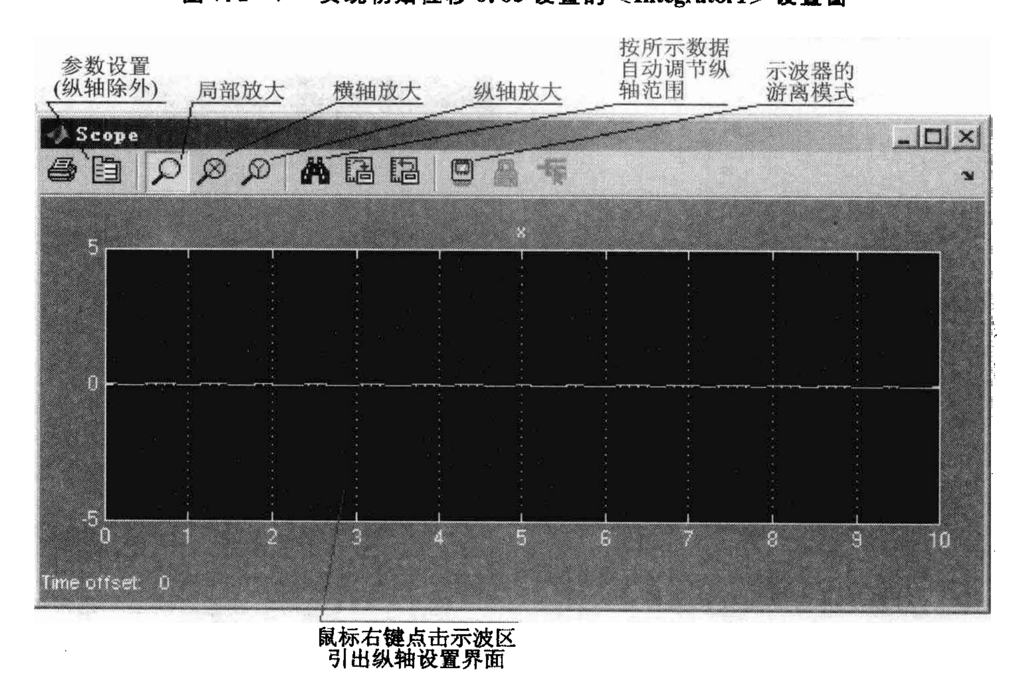


图 7.1-8　坐标范围设置不当时的信号显示

● 单击 Scope 显示窗上的纵坐标范围自动设置图标，示波器显示窗改变为图 7.1－9 所示。在显示窗中，可以看到位移 $x(t)$ 的变化曲线。同时可以发现：纵坐标的适当范围大致在 [－0.06,0.06]；仿真时间取 [0,5] 即可；显示的曲线不够光滑。

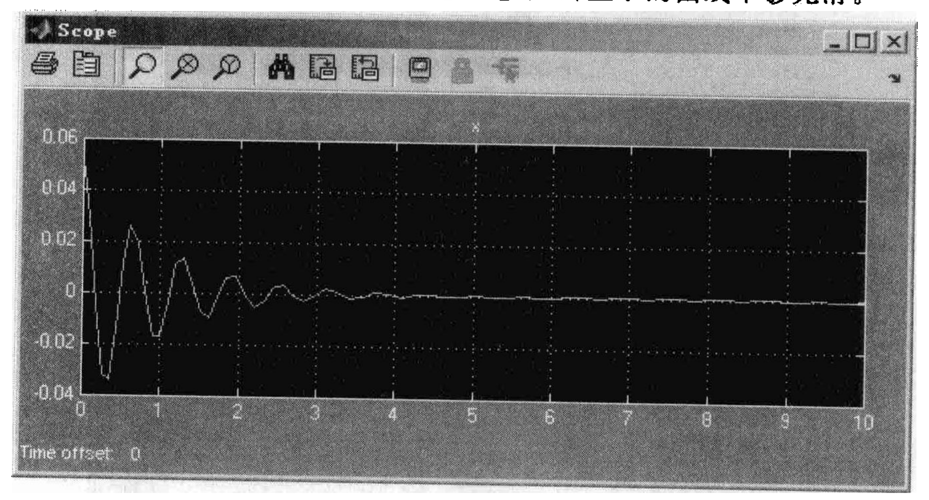


图 7.1－9 采用轴自动设置功能后的信号显示

(12) 据试运行结果，进行仿真参数的再设置

● 示波器纵坐标设置：用鼠标单击示波器的黑色显示屏，在弹出菜单中选择 Axes Properties，引出纵坐标设置对话窗（见图 7.1－10）；把纵坐标的下限、上限分别设置为 (－0.06) 和 (0.06)；单击 [OK]，完成设置。

● 示波器时间显示范围的修改：单击示波器的参数设置图标，引出示波器参数设置窗（图 7.1－11）；在 General 卡片的 Axes 区的 Time range 栏中，填写 5 或 auto；单击 [OK]，完成设置。

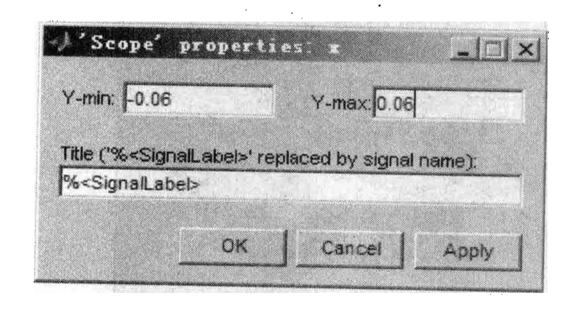


图 7.1－10 对显示屏的纵坐标范围进行设置

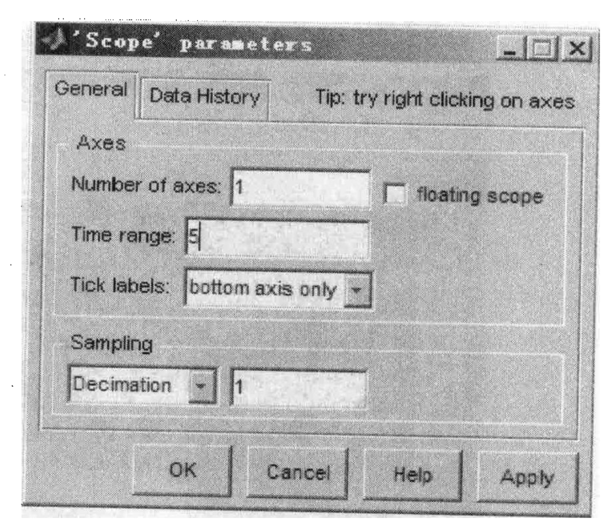


图 7.1－11 对示波器时间显示范围的设置

● 仿真终止时间最简捷的修改方法是：在 exm070101 模型窗“仿真终止时间”栏

10.0 Normal 中的默认值 10 改变为 5。

- 该模型运行后，在指令窗中会出现提示：建议把“最大步长(Maximum Step Size)”修改为 0.2。为此，选中 exm070101 模型窗的下拉菜单项 Simulation\Cofiguration Parameters，引出仿真参数配置窗(见图 7.1-12)；在左侧选择栏中，点中 Solver 项，然后在右侧的 Max step size 栏中，填写 0.2。(注意：在该仿真参数配置窗右上方的“仿真终止时间”栏中，已经显示出上一步修改后的 5。)

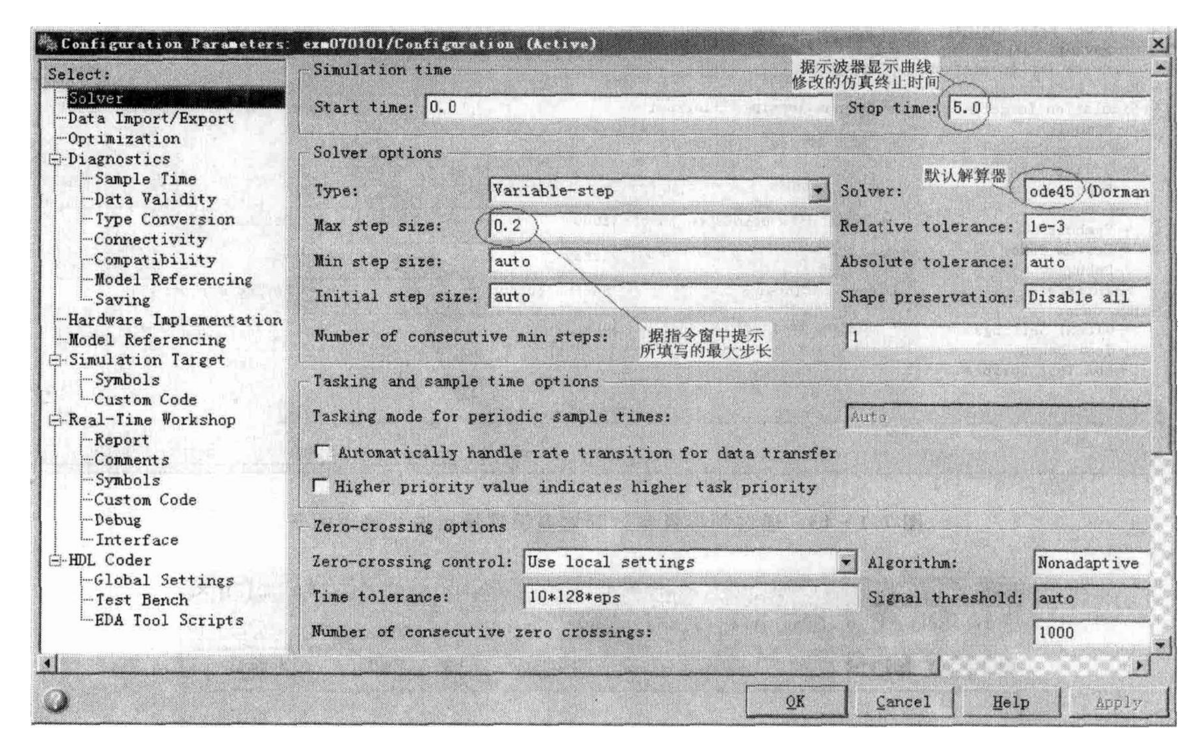


图 7.1-12　通过仿真参数配置窗设置输出曲线光滑因子

- 为使示波器中显示曲线光滑化，再在仿真参数配置窗左侧的选择栏中，选中 Data Import/Export 项，与之相应的参数设置栏便出现在窗口的右侧(参见图 7.1-13)；把右半窗下方 Save options 区中 Refine factor 栏中的默认值 1 改为 5；点击 [OK]键，完成设置。
- 在模型窗(如图 7.1-14)中，x″，x′，x 等信号名称是模型创建者根据需要写入的。标识信号名称的操作方法是：用鼠标双击信号线附近的适当位置后，就会出现一个虚线框；该虚线框中允许输入任何文字。
- 再假如，用户希望看到各模块输出数据的类型，那么应勾选下拉菜单项{Format\Port_signal Dispays\Port Data Types}。完成以上修改后，若再次运行 exm070101，那么可看到：
  - ■ 模型窗(如图 7.1-14)的模块输出口的信号线上会出现 double 字样。这表示该信号是采用“双精度”类型数据运算的。
  - ■ 示波窗中曲线变得比较光滑，并且示波窗的曲线名引用输入信号名 x(参见图 7.1-15)。

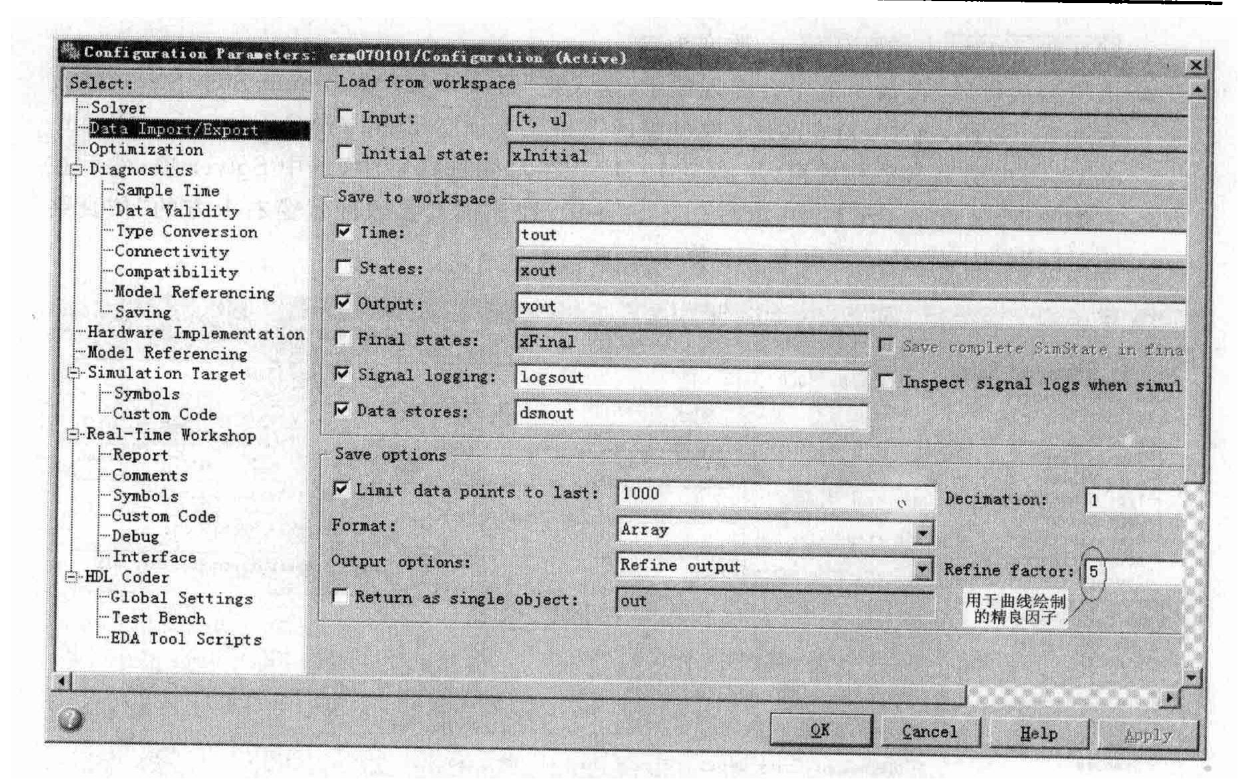


图 7.1-13 通过领导具参数配置窗设置输出曲线光滑因子

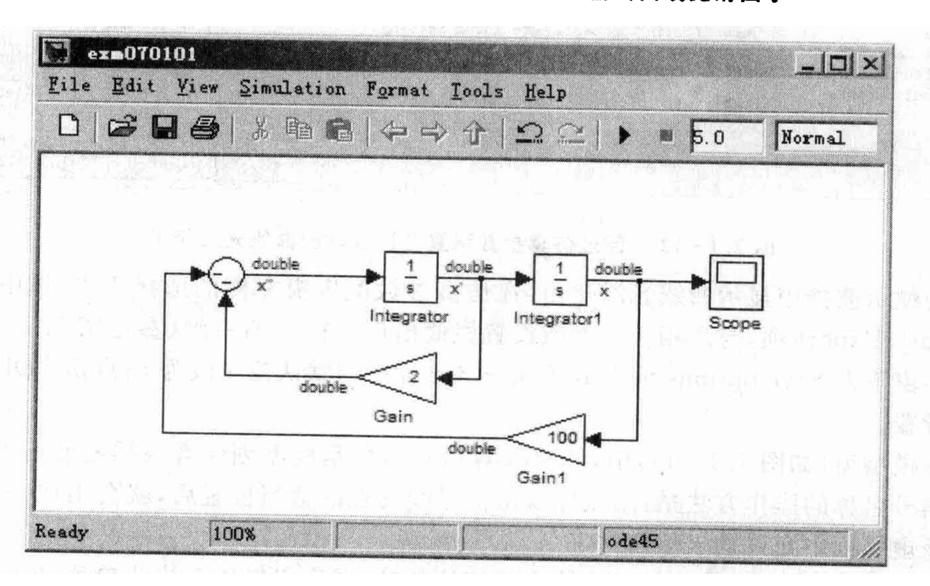


图 7.1-14 仿真参数调整运行后的 exm070101 模型

（13）仿真结果显示

原本比较稀疏的解算器计算数据，经过设定的“光滑因子”下的插值，增加了用于描绘曲线的数据点数，因此使示波器显示出更为光滑的曲线，如图 7.1-15 所示。

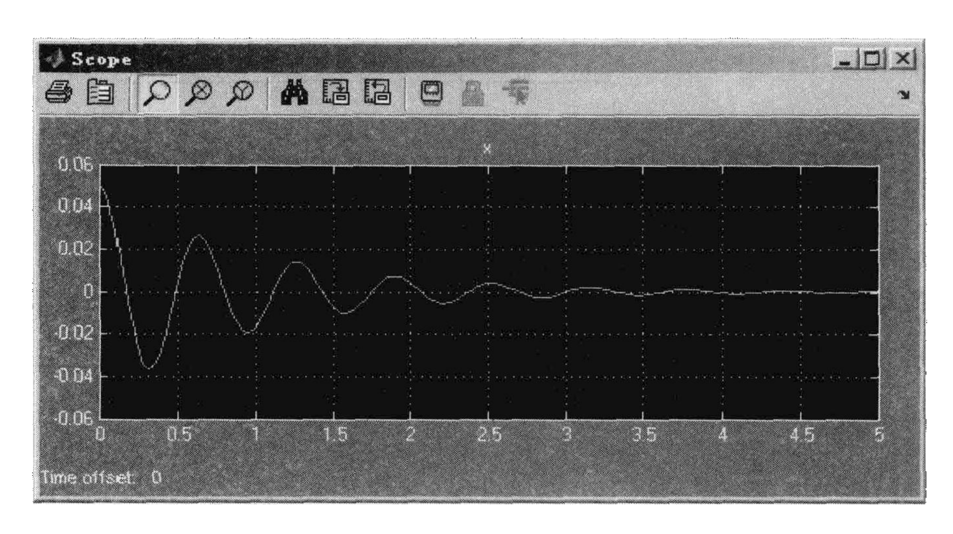


图 7.1-15 适当地显示仿真所得的位移变化曲线

说明

- 本例构建 Simulink 模型的过程是：以“二阶导数”为建模的“起点信号”，然后通过积分模块得到“一阶导数”，再通过积分模块得到“函数本身”。这是基于微分方程创建 Simulink 模型的一般程式。
- 不要企图以“函数”为“起点信号”，通过求导模块得到“一阶导数”，再得到“二阶导数”。在 Simulink 中，求导模块的使用要特别谨慎。
- 本章模型是用 Simulink 7.5 版创建的。由于 Simulink 模块的内码随版本变化较大，所以在其他版本上运行本书提供的 MDL 文件（即 Simulink 模型）可能会遇到困难。但假如读者根据本书算例步骤，在自己的 MATLAB/Simulink 上重新构造模型，那么所得到的仿真结果将是相同的。
- 本例描述的建模过程虽是对“无外力的弹簧—质量—阻尼系统”而言的，但本例勾画的建模程式对更复杂的动态系统（包括机械系统、电路系统等）也都适用，只要那些系统能用微分方程描述。

7.1.2 基于传递函数的 Simulink 建模

例【7.1-2】 对于图 7.1-16 所示的多环控制系统，(1) 求系统传递函数 $G(s)=\dfrac{Y(s)}{U(s)}$；(2) 求该系统的单位阶跃响应。

(1) 建模的基本思路

本算例的系统数学模型是通过形象直观的框图和各环节传递函数给出的。这特别便于采用 Simulink 的传递函数模块建模。

(2) 构造“用于系统传递函数计算”的 Simulink 模型

- 点击 Simulink 模块库浏览器工具条上的图标🗋，引出空白模型窗。
- 从库浏览器复制典型模块。

■ 把＜Simulink\Continuous＞子库的＜Transfer Fcn＞传递函数模块拖进空白模

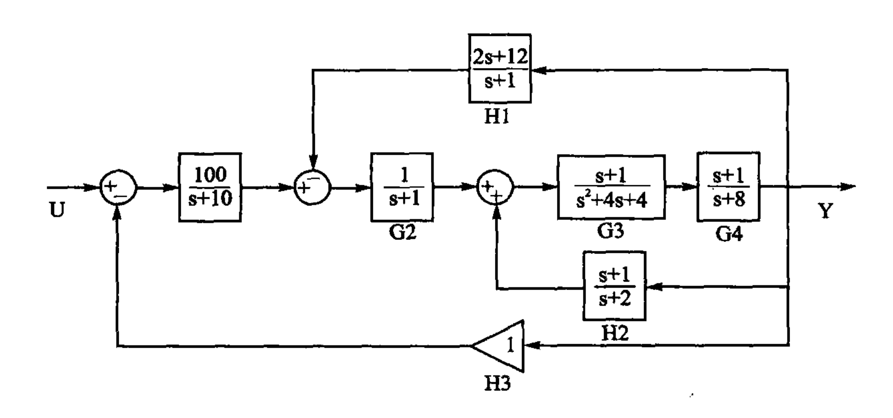


图 7.1-16 多环控制系统

型窗。

- ■ 分别把<Simulink\Common used Blocks>子库的<Gain>增益模块、<In1>输入口模块、<Out1>输出口模块、<Sum>求和模块拖进空白模型窗。

- ● 根据题目给出的要求，在空白模型窗中，进行模块复制。
  - ■ 因为题目需要 6 个传递函数模块，所以需要在空白模型窗中，再复制 5 个传递函数模块。
  - ■ 又因为题目中需要 3 个求和模块，所以需要在空白模型窗中，再复制 2 个传递函数模块。
- ● 对某些模块进行"翻转"操作。
  - ■ 对传递函数中的任何 2 个模块进行翻转操作。用鼠标框选 2 个传递函数模块，比如<Transfer Fcn4>和<Transfer Fcn5>，然后选中下拉菜单项{Format>Flip Block}，就实现了模块的翻转。
  - ■ 再用鼠标点选<Gain>增益模块，选中下拉菜单项{Format>Flip Block}，实现该模块的翻转。
  - ■ 再双击某个求和模块，引出其参数设置对话窗；把符号列表栏(List of signs)中的默认符号(|++)修改成所需的(-+|)；点击［OK］键，完成设置；此时，求和模块的一个"负输入口"就将出现在该模块的上方，如图 7.1-18 所示。
- ● 根据题给框图 7.1-16，整理、排列各模块。注意：在此，把自左向右的模块称前馈通路模块，参见图 7.1-18。
- ● 进行各模块间的连接，参见图 7.1-18。
  - ■ (介绍另一种快捷模块连接法：)用鼠标左键单击<In1>输入口模块；然后按住[Ctrl]键不放，自左至右，用鼠标左键依次单击前馈通路上的各模块，就完成了前馈通路上各模块的连接。
  - ■ 用鼠标左键单击<Gain>模块，再按住[Ctrl]键不放，再单击左边第一个求和模块，就完成<Gain>和第一个求和模块的连接。
  - ■ 采用与上同样的方法，分别完成<Transfer Fcn4>与第二个求和模块的连接，<Transfer Fcn5>与第三个求和模块的连接。
  - ■ 使光标置于<Out1>模块的输入信号线上；按下鼠标右键，光标变为"单线十字叉"；

移动鼠标，使引出的“虚连线”向＜Transfer Fcn4＞模块的输入口靠近；当光标与那输入口靠得足够近，单十字叉变为双十字叉；放开鼠标右键，“虚连线”便变为带箭头的信号连线。

■ 采用与上同样的方法，分别完成“＜Out1＞模块的输入信号线与＜Transfer Fcn5＞输入口连接”、“＜Out1＞模块的输入信号线与＜Gain＞输入口连接”。

● 进行各模块（非几何位置）的参数设置，参见图 7.1－18。

■ 在第一个求和模块参数对话窗的符号列表栏（List of signs）中，把默认符号（|＋＋）修改成（|＋ －）。

■ 双击＜Transfer Fcn2＞模块，引出如图 7.1－17 所示参数对话窗；按照“多项式系数用行向量”表示的规则，在 Numerator Coefficients 分子栏填写［1　1］；在 Denominator Coefficients 分母栏填写［1　4　4］；点击［OK］键，就完成了该模块的传递函数表达。

■ 采用与上同样的方法，分别对各传递函数模块的设置。

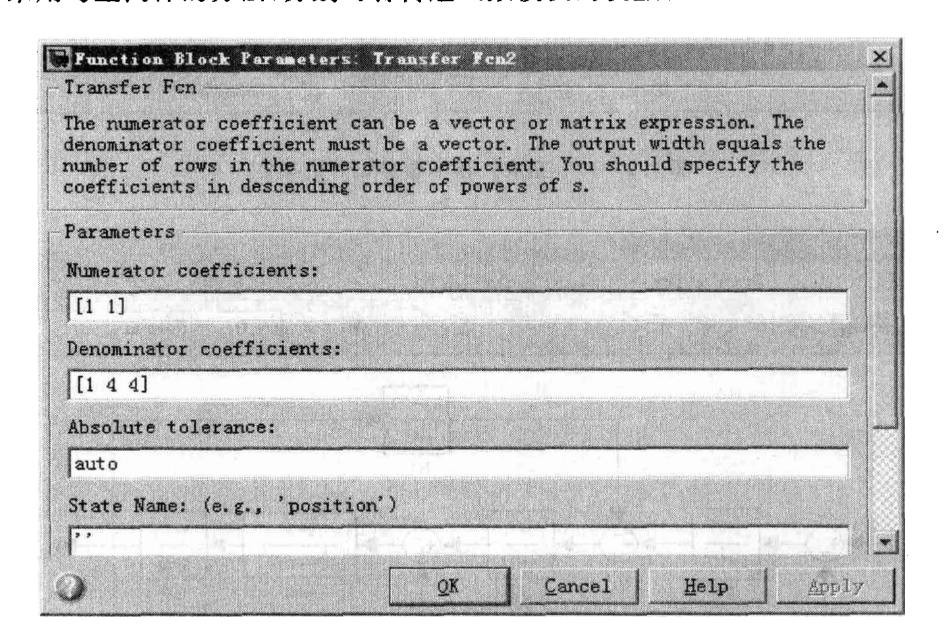


图 7.1－17　传递函数参数设置对话窗

● 模块名称的改写，参见图 7.1－18。

■ 根据题给框图对模型窗中各模块进行如下“名称改写”：

■ 双击＜Transfer Fcn＞模块的默认名，引出文字框；删去原有文字，再写入 G1，即可。

■ 其他模块的名称，可以进行类似的改写。于是得到如图 7.1－19 所示模型。

● 为了避免在指令窗中出现“提示性警告”，还应该在仿真参数配置窗（参见图 7.1－12）左侧选择栏中，点中 Solver 项，然后在右侧的 Max step size 栏中，填写 0.01。

● 经以上操作后，把此模型保存为 exm070102.mdl，参见图 7.1－19。

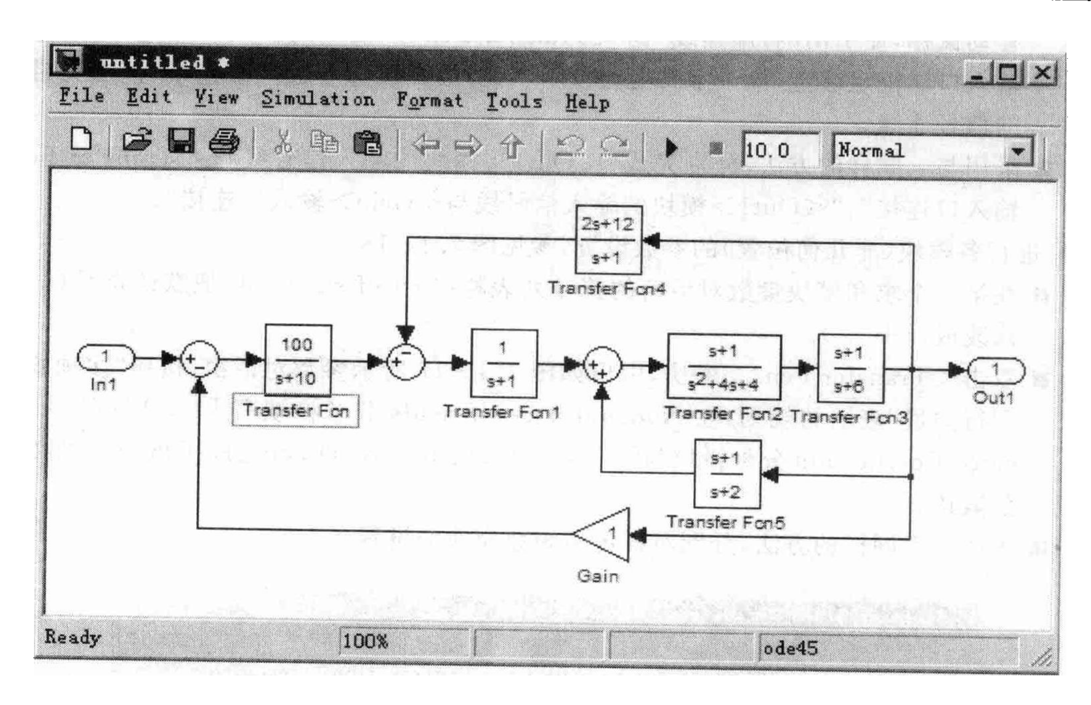


图 7.1-18 完成连接和参数设置的模型

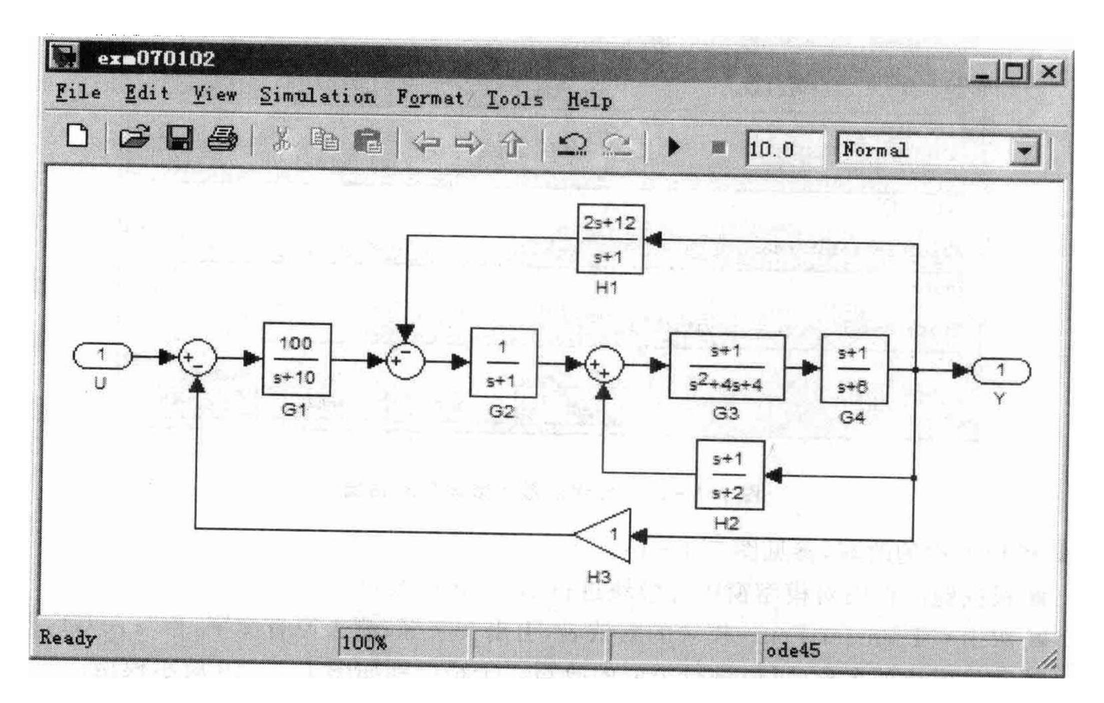


图 7.1-19 用于系统传递函数计算的模型

(3) 系统模型的获取

```
[A,B,C,D] = linmod2('exm070102');          % 从 Simulink 模型得到系统的状态方程
STF = tf(minreal(ss(A,B,C,D)))             % 求状态方程最小实现的传递函数 LTI 对象
[Num,Den] = tfdata(STF);     % 从 LTI 对象中提取传递函数分子分母多项式系数
```

```
Num{:},Den{:}                    % 胞元数组内容显示
2 states removed.

Transfer function:
            100 s^2 + 300 s + 200
-----------------------------------------
s^5 + 21 s^4 + 157 s^3 + 663 s^2 + 1301 s + 910

ans =
         0         0         0  100.0000  300.0000  200.0000
ans =
   1.0e+003 *
    0.0010    0.0210    0.1570    0.6630    1.3010    0.9100
```

(4) 系统的单位阶跃响应(见图 7.1-20)

```
t0 = (0:0.1:5)';                         % 给定待计算响应的时间数据点
[y,t] = step(STF,t0);                    % 求出指定时间点上的输出响应值
plot(t,y,'LineWidth',3)
grid on
axis([0,5,0,0.4])
xlabel('t'),ylabel('y')
```

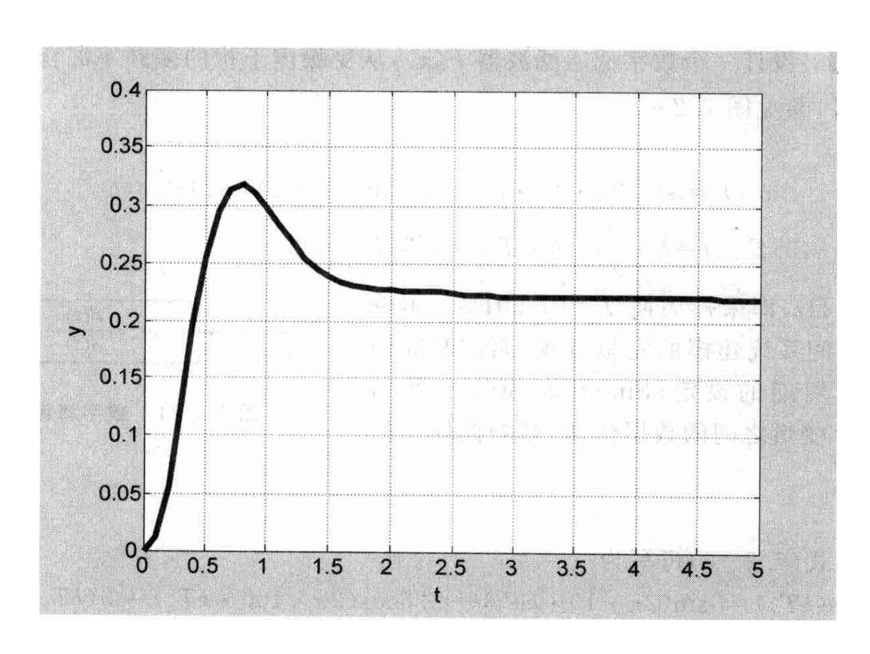


图 7.1-20 系统的单位阶跃响应

说明

- 利用 Simulink 模型,系统传递函数很容易求出。这对复杂系统传递函数的求取特别有用。

- 本例从一个侧面展示了 Simulink 与 MATLAB 之间的交互。
- 指令 ss(A,B,C,D)的含义是：根据获得的状态方程四对组生成一个线性时不变 LTI 模型的状态空间对象(state-space object)。
- 指令 minreal 用于求线性时不变 LTI 对象的最小实现，去除多余的状态变量。
- 经 tf(•)指令的结果 STF 是 LTI 的传递函数对象(transfer functions object)。它直接显示出易读的传递函数。
- [Num, Den] = tfdata (STF)的输出 Num 和 Den 是胞元数组。因此，显示其中内容时，必须采用 Num{ : }和 Den{ : }。
- 系统的单位阶跃响应可以直接通过 step(STF)得到。通过计算得到的线性时不变对象 STF，还可以得到许多其他曲线。如指令 impulse(STF)可以给出系统的冲激响应，bode(STF)给出系统的对数频率曲线等。
- 关于 LTI 对象的详细内容请看 MATLAB 的帮助文件。
- 本例所得传递函数结果与例 2.7 - 2 中的“参数具体化传递函数 Y2Uc”相同。

7.2 离散时间系统的建模与仿真

与连续系统不同，离散时间系统动态过程的数学描述工具是差分方程和 Z 变换传递函数(或滤波器)。在建立离散时间系统的 Simulink 模型时，采样周期是最重要的一个设置参数。本节将通过“低通滤波”算例演示实现离散时间系统建模与仿真的基本要领。

例【7.2 - 1】 设计一个数字低通滤波器 $F(z)$，从受噪声干扰的多频率混合信号 $x(t)$ 中获取 10Hz 的信号，参见图 7.2 - 1。

$$x(t) = \sin(2\pi \cdot 10 \cdot t) + 1.5\cos(2\pi \cdot 100 \cdot t) + n(t) \qquad (7.2-1)$$

在此，$n(t) \sim N(0, 0.2^2)$，$t = k \cdot \frac{1}{f_s} = k \cdot T_s$。采样频率取 $f_s = 1000$ Hz，即采样周期 $T_s = 0.001$ s。本例演示：纯离散时间系统建模的完整过程；离散时间仿真模型中采样周期的设定；Simulink 模块参数与 MATLAB 内存变量之间的数据传递；影响模块几何结构的参数。

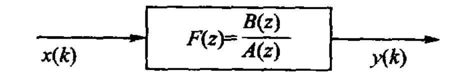


图 7.2 - 1 数字滤波示意图

(1)分析

考虑采样，式(7.2 - 2)可写为

$$x(kT_s) = \sin(2\pi \cdot 10 \cdot kT_s) + 1.5\cos(2\pi \cdot 100 \cdot kT_s) + n(kT_s) \qquad (7.2-2)$$

因为 T_s 是常数，所以式(7.4)可进一步写成

$$x(k) = \sin(2\pi \cdot 10 \cdot k) + 1.5\cos(2\pi \cdot 100 \cdot k) + n(k) \qquad (7.2-3)$$

设无限冲激滤波器有如下形式

$$F(z) = \frac{B(z)}{A(z)} = \frac{b_1 + b_2 z^{-1} + \cdots + b_{n_b+1} z^{-n_b}}{a_1 + a_2 z^{-1} + \cdots + a_{n_a+1} z^{-n_a}} \qquad (7.2-4)$$

那么有

$$y(k) = F(z)x(k) = \frac{B(z)x(k)}{A(z)}$$

$$A(z)y(k) = B(z)x(k)$$

经展开、整理，可写出

$$y(k) = \frac{1}{a_1}[b_1 x(k) + b_2 x(k-1) + \cdots + b_{n_b+1} x(k-n_b) - a_2 y(k-1) - \cdots - a_{n_a+1} y(k-n_a)] \tag{7.2-5}$$

值得指出的是，用 Simulink 模型构建的数字滤波器，就是根据式(7.2－5)进行仿真计算的。这是一种递推计算，是数据的时间流处理。

(2)模型的构建

- 模块的获取
  - 从< Simulink \ Sources >子库，获取< Sine Wave >正弦波模块、< Random Number>随机数模块。
  - 从<Simulink\Math Operations>子库，获取<Sum>或 <Add>求和模块。
  - 从<Signal ProcessingBlockset\Filtering\Filter Implementations>子库，获取<Digital Filter Design>滤波器模块。
  - 从<Simulink\Signal Routing>子库，获取<Mux>合路复用模块。
  - 从<Simulink\Sink>子库，获取<Scope>示波模块。
- 模块的几何参数设置
  - 双击<Add>加运算模块，引出参数对话窗；在 List of signs 符号列表栏中，填写＋＋＋；单击[OK]键，模块就出现三个输入口。
- 按照图 7.2－3 完成各模块间的连接。
- 根据题意，对<Digital Filter Design>数字滤波器设计模块(见图 7.2－2)进行如下设置：
  - 因为题目要求从 10Hz 和 100Hz 两个频率的混合信号中提取 10Hz 信号，所以滤波器采用“低通(Lowpass)”频率响应。
  - 可以实现低通滤波的典型滤波器很多，本例采用“无限冲激响应(Infinite-duration Impulse Response IIR)”的巴特沃思(Botterworth)滤波器。
  - 滤波器的阶数 n_a 取 10 阶。
  - 为了滤去 100Hz 的信号，取截止频率为 30Hz。
  - 题给采样频率 1000Hz 大于被采样信号的最高频率 100Hz 的 2 倍以上，因此这个采样频率对于本例而言是适当的。
- 其他模块的参数设置
  - <Sine Wave>模块的参数设置：Frequency 频率栏，填写 10 * 2 * pi，(注意这里需要填写的是圆频率；Sample time 采样周期栏，填写 1/1000；其余采用默认设置。
  - <Sine Wave 1>模块的参数设置：Frequency 频率栏，填写 100 * 2 * pi，(注意这里需要填写的是圆频率；Sample time 采样周期栏，填写 1/1000；其余采用默认设置。
  - <Random Number>模块的参数设置：Variance 方差栏，填写 0.04，(注意这里是方差，而不是标准差)；Sample time 采样周期栏，填写 1/1000；其余采用默认设置。
  - <Scope>模块的参数设置：纵坐标范围设置为 [－3，3]。

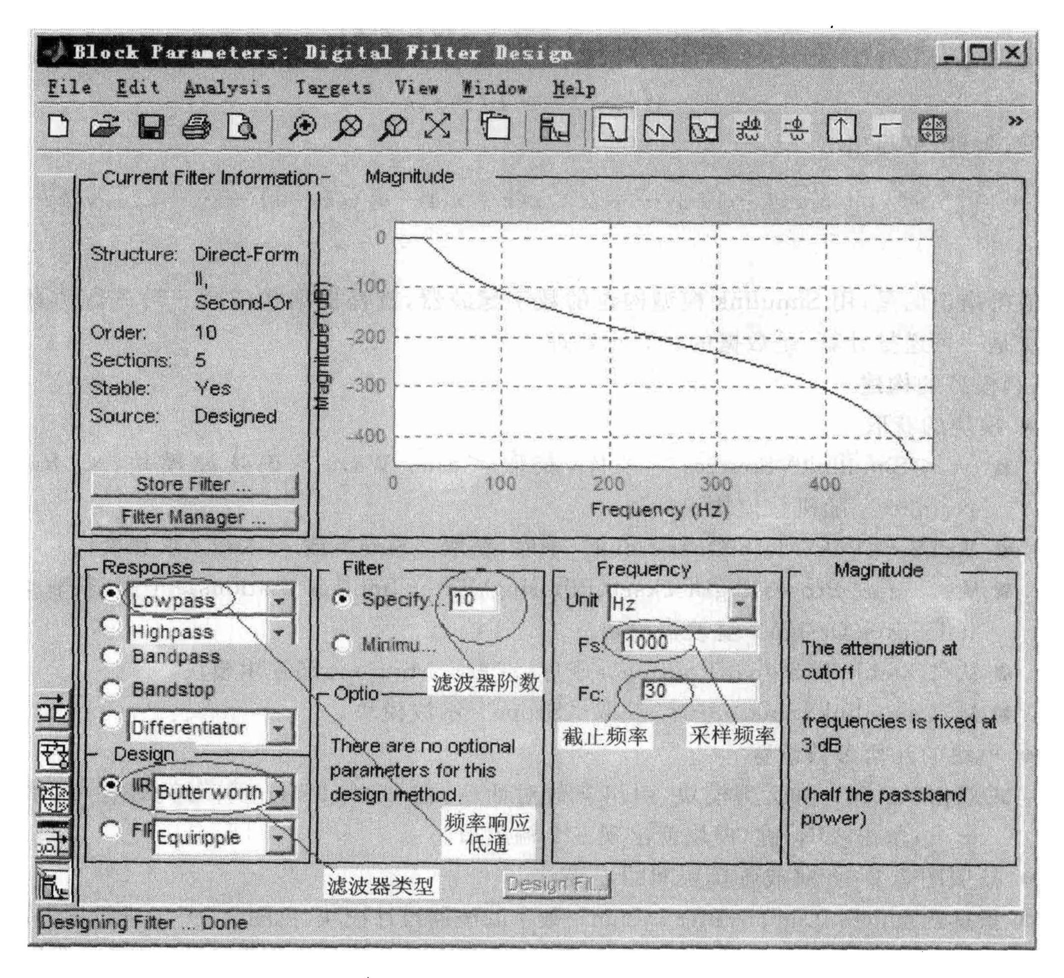


图 7.2-2 数字滤波设计模块参数设置

- 仿真参数配置
  - ■ 选中该模型窗的<Simulation > Configuration Parameters>下拉菜单，引出仿真参数配置对话窗。
  - ■ 在 Stop time 仿真终止时间栏，填写 2。
  - ■ 在 Solver 解算器栏，从下拉菜单中，选择 discrete 离散解算器。
  - ■ 在仿真步长类型 Type 栏，选择 Fixed-step 定步长。这是为便于观察 Simulink 仿真计算的“时间流”特征而“有意”选择的。（注意，一般解算采用变步长。）
  - ■ 在 Fixed-step size 定步长大小栏中，填写 0.0000001。取如此小的步长，也是为了便于读者观察 Simulink 仿真计算的“时间流”特征。
- 把模型保存为 exm070201.mdl。
- 运行模型，在仿真进程中，可见到如图 7.2-4 所示的示波图形。在图中，黄线是混合信号 $x(t)$，红线是经滤波获得的 $y(t)$。

说明

- <Digital Filter Design>模块用在本例显得有些“奢侈”，因为该模块拥有丰富的功能，

可以用于解决远比本例复杂的滤波问题。

- 在对采样离散系统建模时，要保证："采样频率"与"被采样系统中信号最高频率"之比起码要大于 2。在采样频率上限没有特别限制的场合，这比例通常取 5～10。
- 在采样离散系统的 Simulink 模型中，一定要注意各模块的采样频率设置。

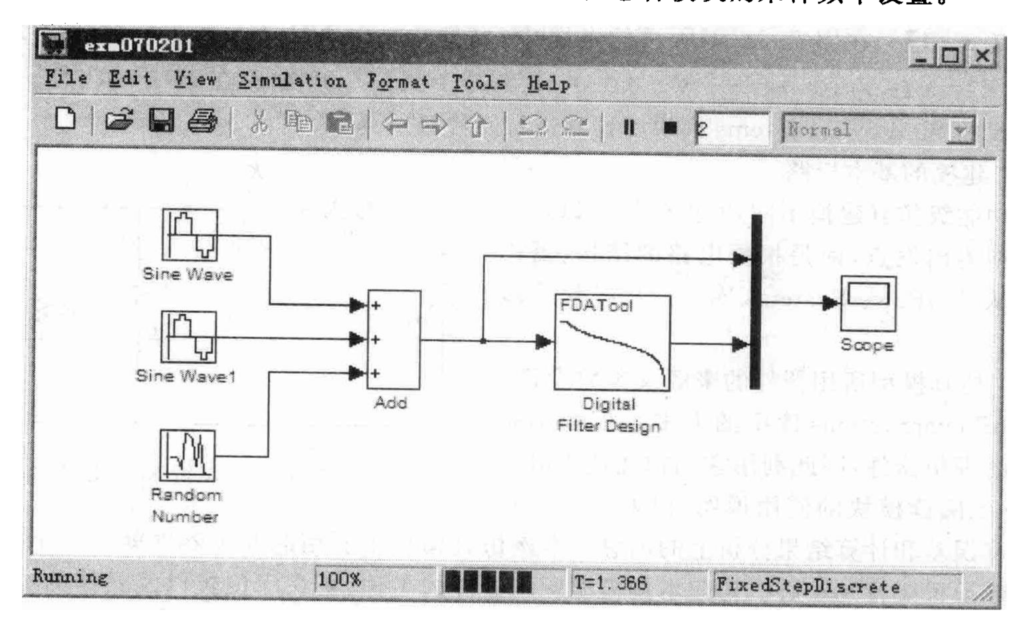


图 7.2-3　仿真进程中的 Simulink 滤波模型

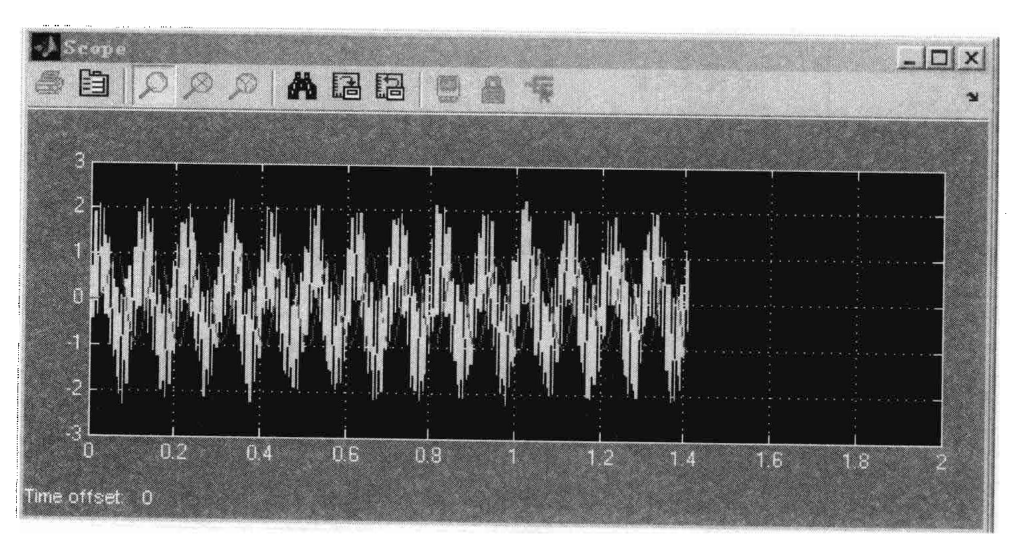


图 7.2-4　仿真进程中的示波图形

7.3 Simulink 实现的元件级电路仿真

就仿真模型逼近被仿真系统的真实程度而言，前两节所建 Simulink 模型属于功能级仿真模型。这种模型所使用的模块与真实的物理器件之间不存在一一对应的关系；这种模型的构

建以抽象了的数学模型为基础。较早的 Simulink，就是进行功能级仿真的软件环境。但最近发布的几个 Simulink 版本，已经在某些专业领域把仿真推进到了元器件级。

本节将利用 SimPowerSystems 中的库模块构建一个在元器件级上对应的电路模型，然后通过该模型进行电量的瞬态分析。

【例7.3-1】 在图 7.3-1 所示的电路中，已知 $L=0.3$ H，$C=0.3$ F，$R_1=2$ Ω，$R_2=0.01$ Ω，$R_3=5$ Ω，$V_C(0^-)=-1$ V，$i_L(0^-)=1$ A，$V_s=10$ V，开关 K 在 $t=0$ 时闭合。试采用 Simulink 的 SimPowerSystems 模块库器件进行元件级仿真，求 i_L 和 V_C。

(1) 建模的基本思路

与功能级仿真建模不同，本例建模不以理论数学模型为出发点，而是根据电路的结构、器件的类型从 SimPowerSystems 库中调用模块直接构建的。

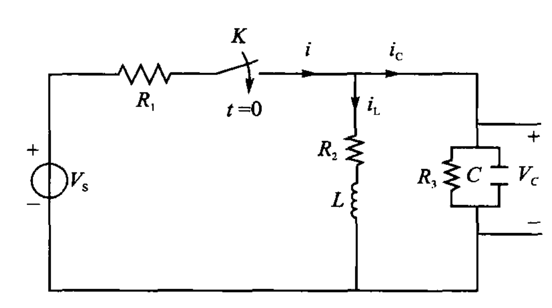


图 7.3-1 二阶 RLC 电路

(2) 仿真模型所用器件的来源及参数设置

SimPowerSystems 库中的大多数模块不是理论上的理想器件，因此利用它们构建理想电路时，要仔细阅读模块的使用说明，以避免出乎意料的仿真误差和计算结果分析上的困惑。本例仿真模型中所用的开关器件就是一个典型，对它的参数设置要特别小心。构建本例模型所用器件的名称、所在库位置以及具体的参数设置见表 7.3-1。

表 7.3-1 建模所需模块的来源和参数设置

模型窗中器件名称	库模块名称	所在库位置	参数设置
直流电压源 V_s	DC Voltage Source	Electrical Sources	● 在 Amplitude 栏，填写 10
电阻 R_1	Series RLC Branch	Elements	● 在 Branch type 栏，选择 R； ● 在 Resistance 栏，填写 2
电阻 R_2 和电感 L 串联支路	Series RLC Branch	Elements	● 在 Branch type 栏，选择 RL； ● 在 Resistance 栏，填写 0.01； ● 在 Inductance 栏，填写 0.3； ● 勾选 Set the initial inductor current，并在 Inductor initial current 栏，填写 1 （注意：该器件朝向必须如图 7.3-2）
电阻 R_3 和电容 C 并联支路	Parallel RLC Branch	Elements	● 在 Branch type 栏，选择 RC； ● 在 Resistance 栏，填写 5； ● 在 Capacitance 栏，填写 0.3； ● 勾选 Set the initial capacitance voltage，并在 Capacitor initial voltage 栏，填写 −1 （注意：该器件朝向必须如图 7.3-2）

续表 7.3-1

模型窗中器件名称	库模块名称	所在库位置	参数设置
开关 K	Breaker	Elements	● 在 Initial state 栏，填写 0； ● 撤消对 External control of switching times 项的勾选；点击[Apply]键；弹出 Switching times 栏，与此同时模块图标上的外控输入口 C 消失； ● 在 Switching times 栏，填写 0；再点击[Apply]键； ● 参见图 7.3-7
电流测量器 Mi	Current Measurement	Measurements	无须设置
电压测量器 Mv	Voltage Measurement	Measurements	无须设置
示波器 Svi	Scope	Simulink\Sinks	● 在示波器对话窗中，将 Number of axes 设置为 2，以产生 2 个输入口； ● 上显示窗的纵轴范围 [0, 7]； ● 下显示窗的纵轴范围 [−2, 3]

(3) 关于仿真模型元器件间连线的说明

SimPowerSystems 仿真模型的构建方法与一般 Simulink 模型的构建方法相似。先把器件模块复制进模型窗；然后对模块进行参数设置以形成元器件的正确几何结构（如通过参数设置，使通用的 RLC 模块成为具体的电阻、电容等）；借助鼠标勾画元器件间的连接；再借助鼠标对元器件和连线进行适当的排列；最后，通过鼠标和键盘操作，对各器件名称进行修改。通过这些操作，就可以获得一个如图 7.3-2 所示的排列整齐、便于观察阅读的仿真模型。

下面以 exm070301 模型（见图 7.3-2）为例，说明构建元器件级仿真模型时的若干特点。

- exm070301 模型中，构成电路的基本器件，如电源、电阻、电容、电感、开关等都只有“无向”端接口。这种端口在模块上以小方块表示。这种端接口仅供器件间的物理连接使用。它既不接受信号输入，也不向外输出信号。
- 构成电路的 RLC、开关等器件的“朝向”决定那器件的“极性”和“正方向”，进而影响“初始状态”设置的正确性，测量数据或曲线的正确性。（涉及“朝向”、“极性”的详细描述请见本例后的说明。）
- exm070301 模型中的示波器<Si>，<Sv>只有“有向”的输入口。这种输入口（输出口）在未经连接的原始模块上以“指向（背向）模块的箭头”表示。它用来显示模型中令人感兴趣的信号。但这种模块无法与只有“无向”端接口的器件直接相连。
- exm070301 中的测量模块<Mi>，<Mv>既有“无向”端接口，又有“有向”的信号输出口。它们是“无向”器件和“有向”模块之间的“中间连接件”。
- 模型中的连线也分两类：一类是物理连接线，它是单纯的线段；另一类是信号流向线，它是带实心箭头的线段。这两种线不允许、也不可能相互连接。
- 假如模型仅由“无向”端口器件组成，那么在仿真过程中就无法观察到仿真模型中任何

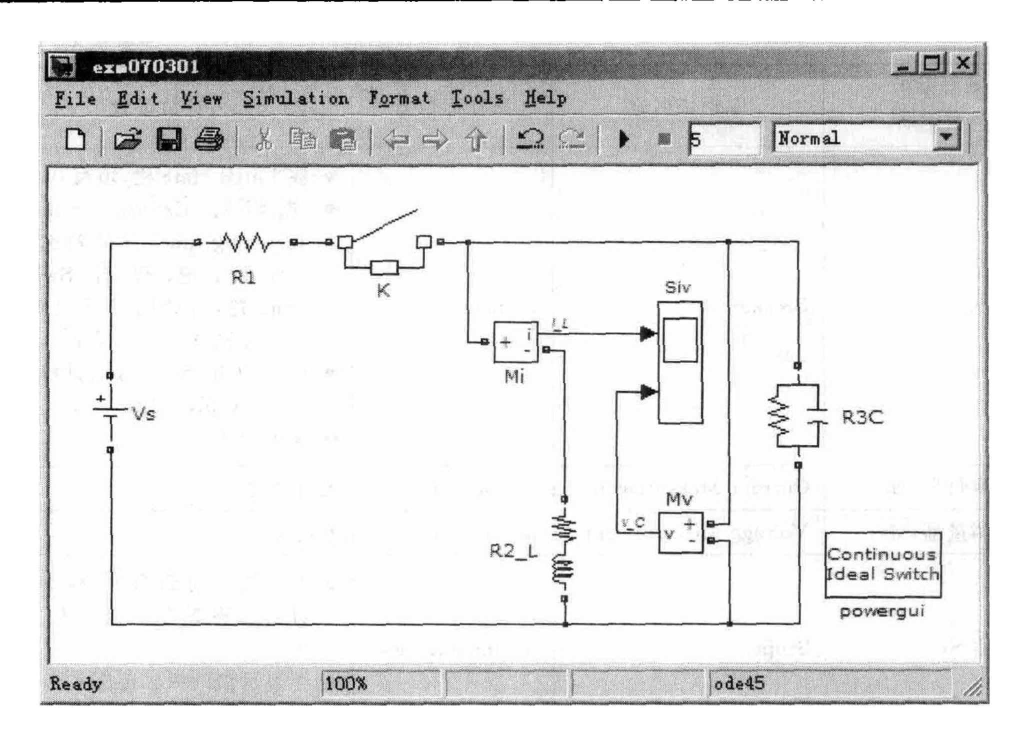


图 7.3-2　元件级仿真模型 exm070301

量的动态变化。

(4) 营造 SimPowerSystems 仿真环境的<powergui>模块

对 SimPowerSystems 构造的模型而言，<powergui>模块是必不可少的。其名称是不可更改的，必须使用 powergui 这个名称。该模块的作用和地位极为重要。因为，正是<powergui>模块的存在，才能把它所在模型窗中的结构模型映射为进行仿真计算的状态方程。

<powergui>模块的母版存放在 SimPowerSystems 库的根目录上。它通常在元器件模型构建完成以后才被复制进那模型窗。<powergui>模块与所构建的元器件模型之间没有任何物理连接，它可以悬浮在模型窗的任何空白处。在 exm070301 模型中，处于模型窗的右下方，参见图 7.3-2。值得指出：刚引入的<powergui>默认库模块的图标上显示 Continuous 字样。

<powergui>模块一般都需根据仿真要求和模型特点进行适当的配置。双击<powergui>模块，就引出如图 7.3-3 所示的仿真电路参数设置对话窗。

(5) 理想开关的设置

SimPowerSystems 是应用性、工程性很强的工具包，因此该工具包中的很多器件都不是“教科书中那种理想器件”。比如，<Breaker>开关模块就有“Breaker resistance(闭路电阻) Ron”、“Snubber resistance(吸收电阻) Rs”、“Snubber capacitance(吸收电容) Cs”、“Switching times(切换时间)”等工程参数项需要设置。

在早先的 SimPowerSystems 工具包中，这种非理想的<Breaker>开关模块是不宜用于直流电路的。为此，在本书的以前版本中，采用把工程参数 Ron 设为 eps，Rs 设为 inf，Cs 设为 0 或 inf 等措施，使仿真得以实现。

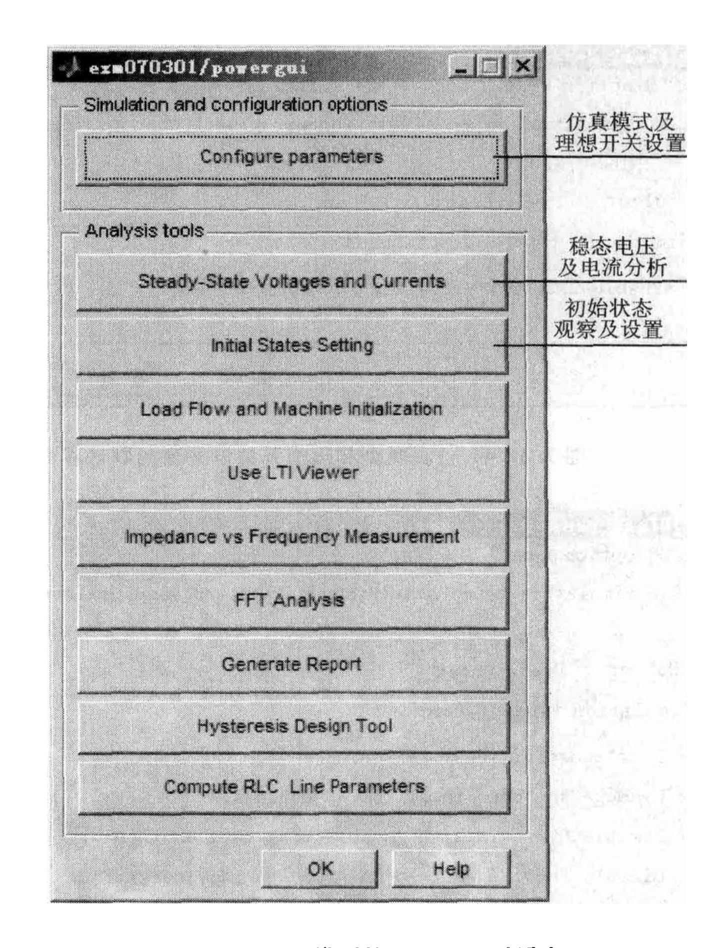


图 7.3－3　exm070301 模型的 powergui 对话窗

在 MATLAB 新版本中，新版 SimPowerSystems 工具包已经具有了把<Breaker>开关模块设置为理想开关的功能。针对本例的具体设置过程如下：

● 单纯的"理想化"设置

■ 点击 Powergui 对话窗上的[Configure parameters]按键，引出 Block Parameters：powergui 对话窗。它的默认界面如图 7.3－4 所示，其特点是：

在该界面的 Solver 页上，Simulation type 仿真解算器类型栏，默认选项为 continuous。

Enable use of　ideal switching devices 理想化选择栏，默认设置是"不勾选"。为开关理想化，推荐勾选此项。注意：此项的勾选，将引出新的选项(参见图 7.3－5)。

■ 点击[Apply]，确认"理想化"设置。此后，exm070301.mdl 模型窗中的<powergui>模块图标上的字样将改变为"Continuos"和"Ideal Switch"(参见图 7.3－2)。

■ 应该说，经过以上操作的模型已经解决了开关的理想化问题。但是，还存在两个问题：

假如双击模型窗中的<Breaker>模块，而引出的对话窗。那么，该对话窗中各项工程性参数的设置仍能破坏以前的"理想化"设置。

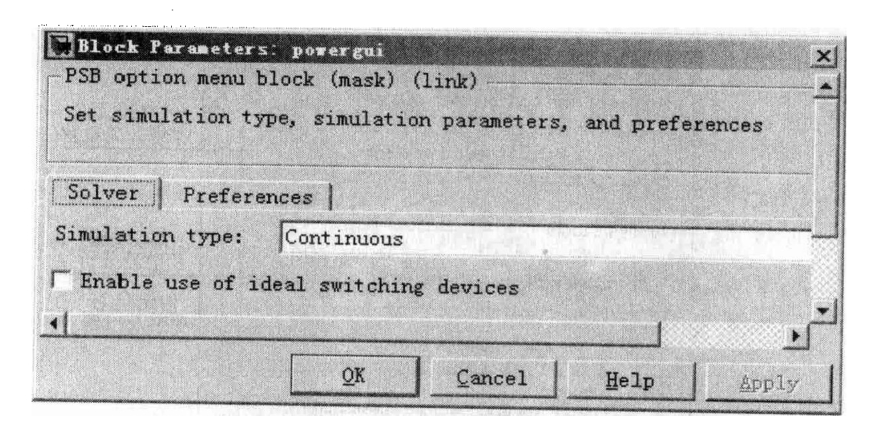


图 7.3-4　仿真类型和理想开关设置窗的默认界面

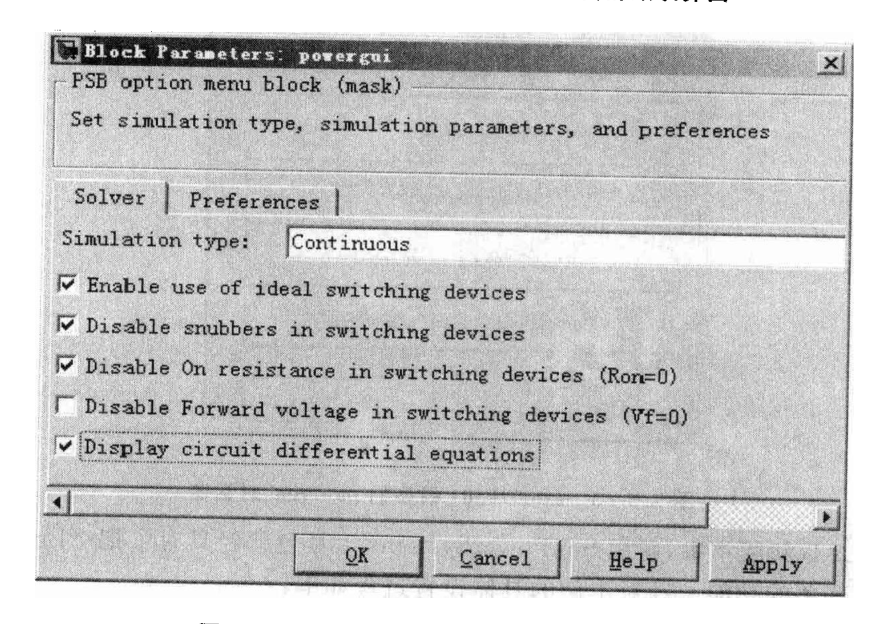


图 7.3-5　仿真模式及理想开关设置对话窗

在模型窗中，<Breaker>模块的图标仍带有“吸收阻抗 Snubber”器件。参见图 7.3-2。

- 更恰当的“理想化”设置

当在 Block Parameters：powergui 对话窗中，勾选 Enable use of ideal switching devices 项以后，对话窗界面上就会出现 4 个新的选项，如图 7.3-5。

■ Disable snubbers in switching devices 选项。

若勾选此项，点击[Apply]键，再选中模型窗下拉菜单{Edit>Update Diagram}，那末引起以下变化：

模型窗中<K>模块图标上不再有并联的 Snubber 吸收阻抗器件(参见图 7.3-6)；

模型窗中<powergui>模块上的显示字样也随之变化(参见图 7.3-6)；

在<K>开关模块对话窗中，与吸收阻抗有关的 Snubber resistance Rs 栏和

Snubber capacitance Cs 栏都被“失能的灰化”(参见图 7.3－7)。

图 7.3－6　理想开关设置后的模型窗

■ Disable On resistance in switching devices (Ron＝0)选项。

若勾选此项，点击[Apply]键，选中模型窗下拉菜单{Edit>Update Diagram}。那末开关的闭路电阻 Ron 将被设置为 0。并因此，<K>开关模块对话窗的 Breaker resistance Ron 栏也将呈现为“失能的灰化状态”(参见图 7.3－7)。

■ Disable Forward voltage in switching devices (Vf＝0)选项。

Vf 为强迫换向装置或整流器的正向电压。本例不涉及。

■ Display circuit differential equations 选项。

若勾选此项，点击[Apply]或[OK]键，那末运行 Simulink 模型后，将在 MATLAB 指令窗中，输出相应的被仿真电路的动态微分方程。

● 关于“开关模块对话窗”的进一步说明

经以上操作后，在开关模块 K 的对话窗中，有三个对话栏已经“灰化”，这意味着：原先的默认值已经因“此前的理想化操作而失去效用”(参见图 7.3－7)。这可有效防止“那些栏目被再次设置”。

在此对话窗中，还有两个栏目值得进一步说明。

■ 开关初始“开、闭”状态 Initial state 栏的含义：

该栏若填写 0，则表示仿真启动前，开关处于“开启”状态。

该栏若填写 1，则表示仿真启动前，开关处于“闭合”状态。

■ 开关切换时间 Switching times 栏的功用

该栏仅当“内控模式”时才出现。换句话说，只有在 External control of switching times“不勾选”时才存在。

该栏可以填写“行数组”，(标量可看作只有一个元素的行数组)。

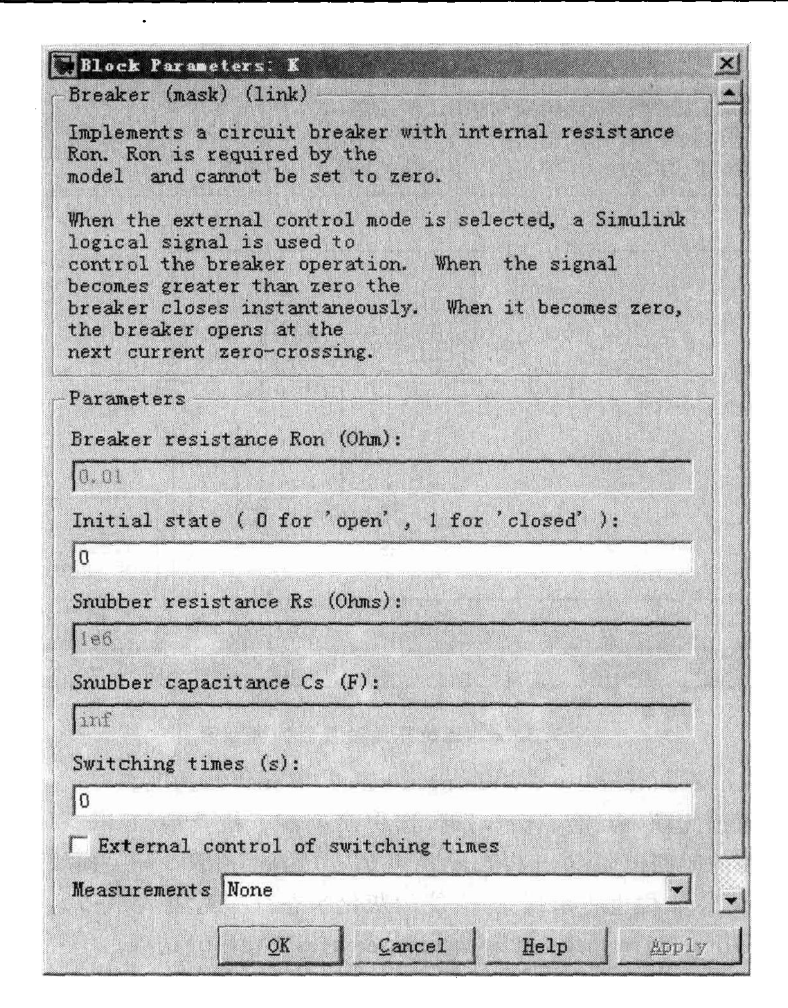


图 7.3-7 开关经失能设置后的对话窗

在行数组给定的每个"时刻",开关将从"原先开闭状态",比如"开启"状态,瞬间转换为"另一个对立的开闭状态",即相应为"闭合"状态。

在本例中,该栏应填写一个"标量"0,用以表示,开关的切换动作发生在仿真启动的那一瞬。

(6) 仿真类型的选定

双击<powergui>模块,就引出如图 7.3-3 所示的仿真电路参数设置对话窗;再点击 Powergui 对话窗上的[Configure parameters]按键,引出如图 7.3-4 所示的 Block Parameters: powergui 对话窗。

在 Block Parameters: powergui 对话窗中,Solver 页的 Simulation type 项有三个选项:

● Continuous 选项

把模型窗中的电路映射为连续(微分方程)模型。至于具体采用哪种解算器,最大步长、相对误差等还需要在仿真参数配置窗中进行设置。

因为带开关或功率电子开关器件的电路,大多映射为"刚性方程",因此解算器,宜采用 ode23tb;相对容差设为 1e-4;最大步长可取 auto,或根据模型试运行后,指令窗

中出现的提示进行修改。

● Discrete

把模型窗中的电路映射为离散(差分方程)模型。

● Phasor

把模型窗中的电路映射为相量模型，应用于交流电路的稳态分析和仿真。

● 表 7.3-2 中所列的三组“解算器及参数设置”都能应用于本例。

表 7.3-2　解算器及仿真参数设置

映射模型	形式	Continuous 连续(默认)		Discrete 离散
	采样周期 T_s			不大于 0.01s
解算器	算法	Ode23tb	ode45	discrete
	最大步长	0.1	0.1	不小于
	相对误差	0.0001	0.001	0.001

(7) 模型初始状态的设定

SimPowerSystems 模型的初始状态设定有两个途径：一，在元器件模块上设置；二，在<powergui>中设置。这两种设置方式是关联的，也就是说，在一种方式中设置的初始值将相应地表现到另一种设置界面中。

值得强调：本例在器件对话窗中把电感、电容初始状态值设置为+1、-1 时，“R2,L”、“R3,C”器件的朝向都应如图 7.3-2 或 7.3-6 所示。更具体地说，“R2,L”和“R3,C”的“器件朝向”都是“向下(Down)”的。这可以通过以下步骤鉴定：分别选中“R2,L”和“R3,C”；在指令窗中运行指令 get_param(gcb,'Orientation')；该指令两次运行结果都应为 down；由此表明“R2,L”器件的电流正方向是“向下”，“R3,C”器件上的电压是“上正下负”。(更详细的描述，参见本例后的说明及表 7.3-3)。

由于本例初始状态已经通过“器件对话窗”设置，所以无需在<powergui>中再次对初始状态进行设置。此时，点击<powergui>模块对话窗(参见图 7.3-3)中的“初始状态设置(Initial States Setting)键”，就可以从弹出的图 7.3-8 所示的对话窗窗左边的“状态变量列表(Initial electrical state values for simulation)”中，看到元器件模型中相应储能支路上已经设置的初始状态值。注意：所显数值的“正、负”是以现有器件的“朝向”为参照的。

通过右侧的“To Steady State”或“To Zero”选项，可以把电路中的状态强迫设置为“稳态值”或“零初始”，而不管此前电路状态如何设置。

此外，右侧还有两个“状态重载(Reload states)”选项：“取自文件(From File)”和“取自电路模型(From Diagram)”，使再次仿真时的初始状态可取自“先前保存的模型状态文件”或“取自该电路的状态当前值”。

(8) 仿真运行

经过以上各步的正确设置后，就可以点击 exm070301 模型窗上的仿真启动键，并得到如下仿真结果，参见图 7.3-9。注意，图中仿真曲线是在电感电流初值为(1)、电容电压初值为(-1)的情况下获得的。

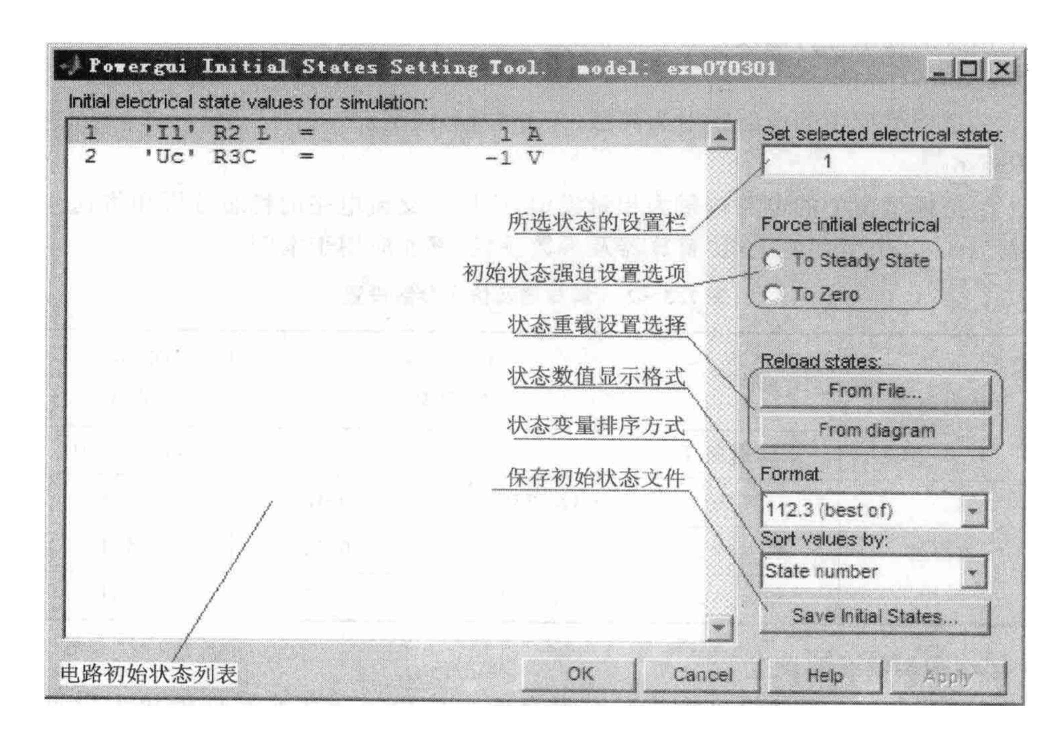


图 7.3-8 powergui 的初始值设置窗

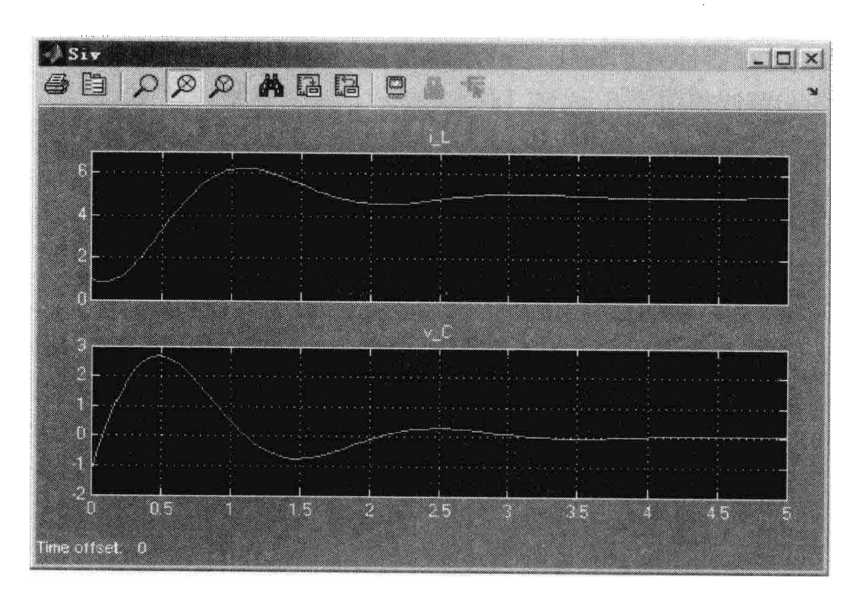


图 7.3-9 仿真所得的电感电流和电容电压变化曲线

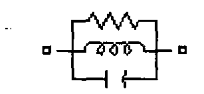
说明

- 器件(模块)朝向和电压极性、电流正方向的关系
  - 当对电容电压,或电感电流设置非零初始值时,会遇到"这些物理量的极性认定问题";当使用万用表(Multimeter)测量支路电压或电路时,会遇到"这些物理量的正方向假设问题"。

■ 在 SimPowerSystems 中，器件（模块）的极性（Polarities）和正方向（Positive Direction）是由该模块的"朝向"（Block Orientation）决定的。



Series RLC Branch



Parallel RLC Branch

图 7.3－10　RLC 库器件原型

对于库器件（模块）来说，"水平放置器件（Horizontal Blocks）"的默认朝向是"向右（Right）"，而"垂直放置器件（Vertical Blocks）"的默认朝向是"向下（Down）"。比如，本例所调用的"串联 RLC（Series RLC Branch）"和"并联 RLC（Parallel RLC Branch）"库器件都是"水平放置器件"，如图 7.3－10 所示：

这两个库器件的极性规定是：左端为"正"，右端为"负"。换句话说，支路的正方向规定是："由左向右"。

■ 在建模过程中，库器件（模块）的朝向，可能因反转、旋转等操作而改变。在此情况下，判断模块的朝向的最可靠方法是：用鼠标选中所需判断极性的器件；然后在指令窗中运行指令 get_param(gcb,'Orientation')；该指令运行结果给出"模块的朝向"；再据此"朝向"从表 7.3－3 中就可以查到正确的"极性"和"正方向"标定。

■ 模块朝向的辅助识别法：

假如串联 RLC 器件由电阻和电容（电感）共同组成，那么该器件的朝向可据"电容（电感）相对电阻的位置确定"。电容（电感）在电阻右（或下）侧，则器件朝向右（或下）。

假如 RLC 器件中，包含电容，那末可据"构成电容符号的平直线和弯弧线的位置"确定器件朝向。"弯弧线"在"平直线"右（或下）侧，则器件朝向右（或下）。

表 7.3－3　器件（模块）朝向所对应的电流、电压正方向

模块朝向	电流正方向	电压
right	left→right	$V_{left}-V_{right}$
left	right→left	$V_{right}-V_{left}$
down	top→bottom	$V_{top}-V_{bottom}$
up	bottom→top	$V_{bottom}-V_{top}$

● 影响仿真初始状态的三个途径

■ 在模型窗各种储能器件各自的对话窗中，有专门对本器件初始状态进行设置的栏目。

■ 由<powergui>模块对话窗上[Initial States Setting]按键引出的如图 7.3－8 所示的对话窗，可以对被仿真电路中所有器件的初始状态进行设置。

■ 在如图 7.3－5 所示的 Block Parameters：powergui 对话窗的 Preferences 页上，有个对模型器件初始状态进行强制设置的栏目 Start simulation with initial electrical states from，该栏的默认选项是 blocks。

习题 7

1. 利用 Simulink 求解 $I(t)=\int_0^t e^{-x^2}\,dx$ 在区间 $t\in[0,1]$ 的积分，并求出积分值 $I(1)$。（提

示：时间变量由 Clock 产生；注意使用 Product，Math function，Integrator，Display，Scope 等库模块；计算结果可与例 4.1-5 对照。）

2. 利用 Simulink 求解微分方程 $\frac{d^2x}{dt^2}-\mu(1-x^2)\frac{dx}{dt}+x=0$，方程的初始条件为 $x(0)=1$，$\frac{dx(0)}{dt}=0$。在增益模块"Gain"取值分别为 2 和 100 的情况下（即数学表达式中 $\mu=2$，$\mu=100$）运行，给出运行结果。（提示：注意使用 Constant，Product，Add，Gain，Integrator，Scope 等库模块；注意初始状态设置；针对不同 μ，采用不同解算器，并设置不同仿真终止时间；运算结果可与例 4.1-9 对照。）

3. 已知某系统的框图如图 p7-1 所示，求该系统的传递函数。（提示：参照例 7.1-2。）

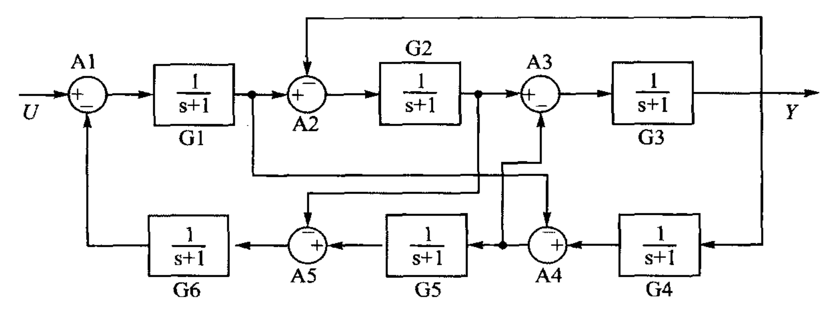


图 p7-1

4. 采用 Simulink 基本库和 Signal Processing Blockset 库的"连续"时间的模块构建的 Simulink 模型解决第 7 章算例 7.2-1。（提示：可使用 Sine Wave，Analog Filter Design 等库模块；注意采样时间设置。）

5. 在如图 p7-2 所示的交流电路中，其中 $Z_1=j1(\Omega)$，$Z_2=Z_4=Z_5=1(\Omega)$，$Z_3=-j1(\Omega)$，$\dot{V}=20\angle120°(V)$，$\dot{I}=10\angle45°(A)$，$f=50(Hz)$，求 Z_3 支路中的电流和 Z_5 两端的电压。（提示：采用相量分析法；要注意电压源、电流源库模块的频率、相角设置；特别要注意模块所采用的幅值是"峰幅值"，它应是"有效值"的 $\sqrt{2}$ 倍；电压、电流测量模块的输出信号，可选择 Magnitude-Angle 幅相模式；powergui 模块的仿真类型 Simulation type 应选择"相量 Phasor"；仿真终止时间非 0 即可；仿真可采用纯离散解算器 discrete。）

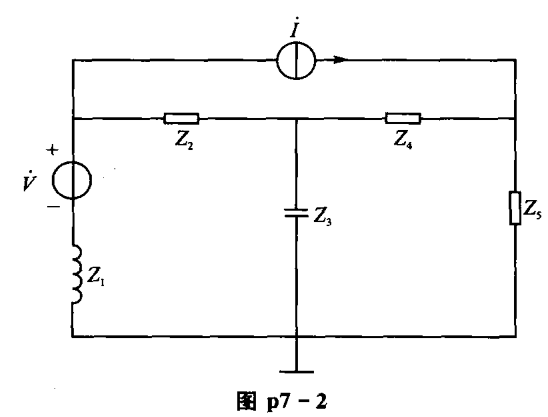


图 p7-2

第8章 图形用户界面（GUI）

假如读者所从事的数据分析、解方程、计算结果可视工作比较单一，那么一般不会考虑图形用户界面(Graphical User Interfaces,GUI)的制作。但是如果读者想向别人提供某种新的设计分析工具，想体现某种新的设计分析理念，或想进行某种技术、方法的演示，那么 GUI 应该是很好的选择。

MATLAB 本身提供了很多的 GUI，如 sisotool（单输入单输出控制系统设计工具）、fdatool（滤波器设计和分析工具）等。这些工具的出现不仅提高了设计和分析的效率，而且改变了原先的设计模式，引出了新的设计思想，改变了或正在改变着人们的设计、分析理念。正是出于这种观察，在此将 GUI 内容辟为一章列入本教程。

本章由 4 个精心设计的算例组成。这 4 个算例，一方面尽可能多地向读者展现构成 GUI 的各种控件或组件，另一方面借助算例中回调函数的编写，较快地将读者引向 GUI 制作的纵深。相信读者只要耐心地按照示例进行操作，便能感受 GUI 成功制作的愉快，同时事半功倍地掌握 GUI 的制作技巧。

8.1 GUI 入门示例

假若读者能循本节算例指引，亲自动手一步步地操作，就一定能相当快地了解创建图形用户界面的步骤和制作要领，感受到自己成功制作 GUI 的快乐。

例【8.1－1】 为演示归一化二阶系统 $G(s)=\frac{1}{s^2+2\zeta s+1}$ 中阻尼比 ζ 对单位阶跃响应的影响，需要制作如图 8.1－1 所示的用户界面。要求：在界面右侧的编辑框中输入阻尼比 ζ 的具体“大于 0 的数值标量”，就能在坐标轴上显示出相应曲线。本例目的：(A)不管是否熟悉 GUI 界面编辑器，也不管是否熟悉图形句柄，先循本例步骤操作，尝试创建用户界面；(B)感受创建用户界面的基本步骤；(C)初步了解 GUI 界面编辑器开发环境；(D)大致了解用户界面的工作原理；(E)初步了解轴、静态文本框、可编辑文本框等三种组件的使用。

(1) GUIDE 开发环境的引入

由于本例所需创建用户界面的外观要求和动作要求都十分清晰，因此可以直接进入界面设计阶段。

为引出 GUIDE 开发环境，在指令窗中运行 guide 指令，就会出现如图 8.1－2 所示的对话窗。该窗有两个页面：“新建 GUI(Create New GUI)”和“打开 GUI(Open Exisiting GUI)”。

在“Create New GUI”页面，先点选“Blank GUI (Default)”条目，再点击[OK]键，引出

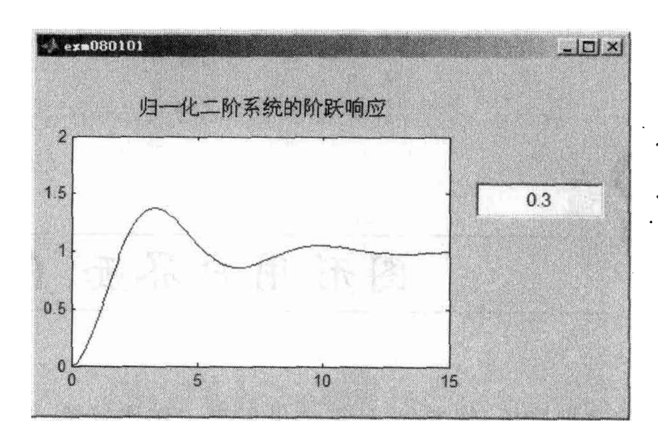


图 8.1-1 待制作的二阶系统单位阶跃响应演示界面

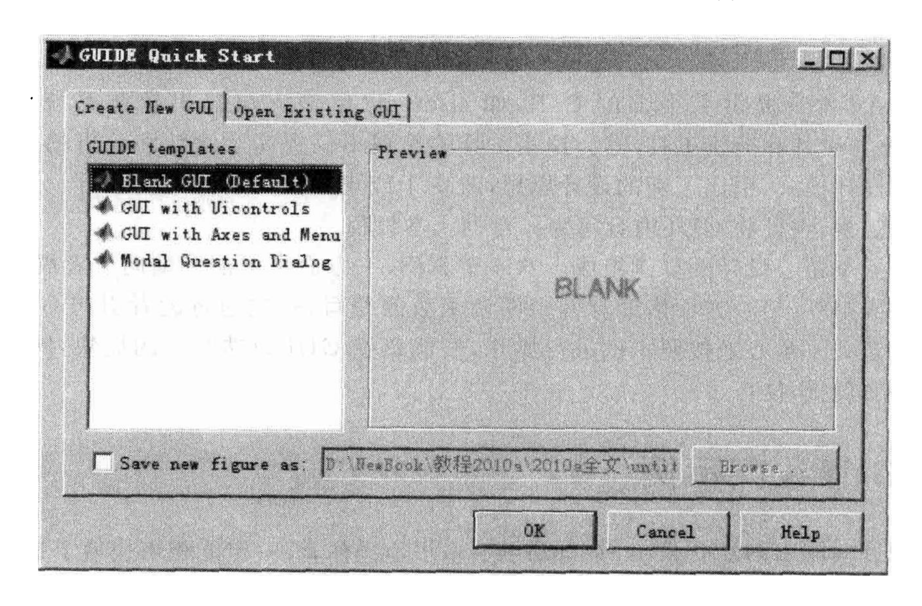


图 8.1-2 进入 GUIDE 开发环境的默认引导对话窗

GUI 空白界面编辑器(GUI Layout Editor),如图 8.1-3 所示。

顺便指出:图 8.1-3 左侧的“控件及组件”所呈现的是默认外形。假若想使控件、组件图标带其名称一起显示,应采用以下步骤设置:

- 选中界面编辑器菜单项{File > Preference},引出 GUIDE 选项设置界面;
- 在出现对话窗右侧,勾选“Show names in component palette”项,并点击[OK]键,设置结果如图 8.1-4 左侧所示。

(2) 根据要求选配界面组件

一般说来,在利用 GUI 界面编辑器设计应用界面前,应首先根据任务要求绘制应用界面的草图。由于本例的设计界面简单、清晰,要求明确,所以无须再进行界面的轮廓草绘,可以直接进入界面制作阶段。

- 通过操作位于 GUI 设计界面右下角的“小黑块”,即“界面缩放句柄”,把设计界面编辑器(Layout Editor)的长宽调整到希望的大小。注意:界面大小也就是未来应用界面出

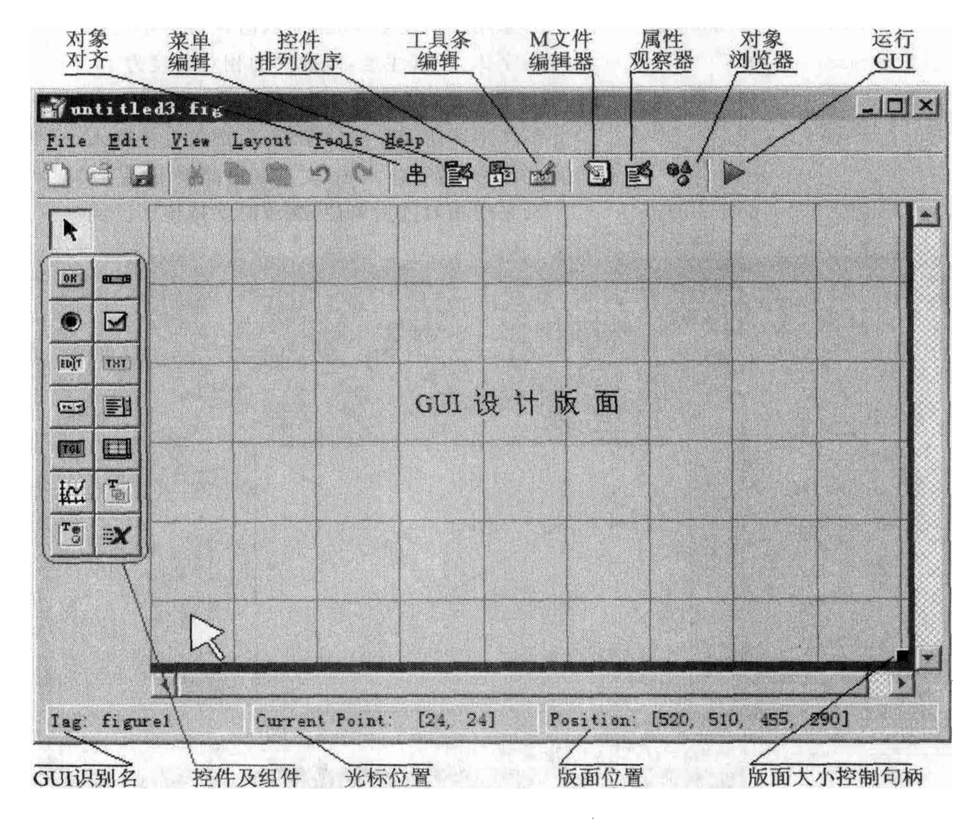


图 8.1-3 默认的空白用户界面开发环境 GUIDE

现时的"缺省大小"。

- 在"组件模板区",点选"轴"图标 Axes ;然后在工作区中的适当位置,拉出适当大小的"轴位框",以供绘制响应曲线使用。参见图 8.1-4。
- 点选"静态文本框"图标 Static Text ;通过鼠标的拖拉操作,把该文本框的设置在"轴位框"的上方;文本框的大小可以用鼠标调节。参见图 8.1-4。
- 点选"可编辑文本框 "图标 Edit Text ;用鼠标把它设置在"轴位框"的右侧。参见图 8.1-4。

(3) 界面组件的参数设置

- 用户界面窗的参数设置:
  - ■ 双击设计工作区"空白处",引出"窗属性编辑器"进行用户界面窗的参数设置,如图 8.1-5所示。再点选设计工作区,点击界面编辑器工具条上的图标,出"属性编辑器"。
  - ■ 在"窗属性编辑器(Inspector:figure)"中,设置下列属性值:

```
Resize      on            %该设置很重要。它决定图形窗是否可缩放。
Units       normalized    %采用"归一化"单位计量窗口大小
```

注意:%符后的文字是作者所加的说明,而非对话框中填写内容,以下均同。

- 双击窗左侧上方的"静态文本(Static Text)"控件(Controls),打开其属性编辑器,作下列属性设置:

FontUnits	normalized	%采用相对度量单位，缩放时保持字体比例
FontSize	0.5	%字体大小(注意：该控件的相对高度为 1)
String	归一化二阶系统的阶跃响应	
		%文本区所需显示的字符，由键盘直接输入
		%注意：若字体太小，文本区可能看不到字符显示
Units	normalized	%采用相对度量单位，缩放时保持该区比例

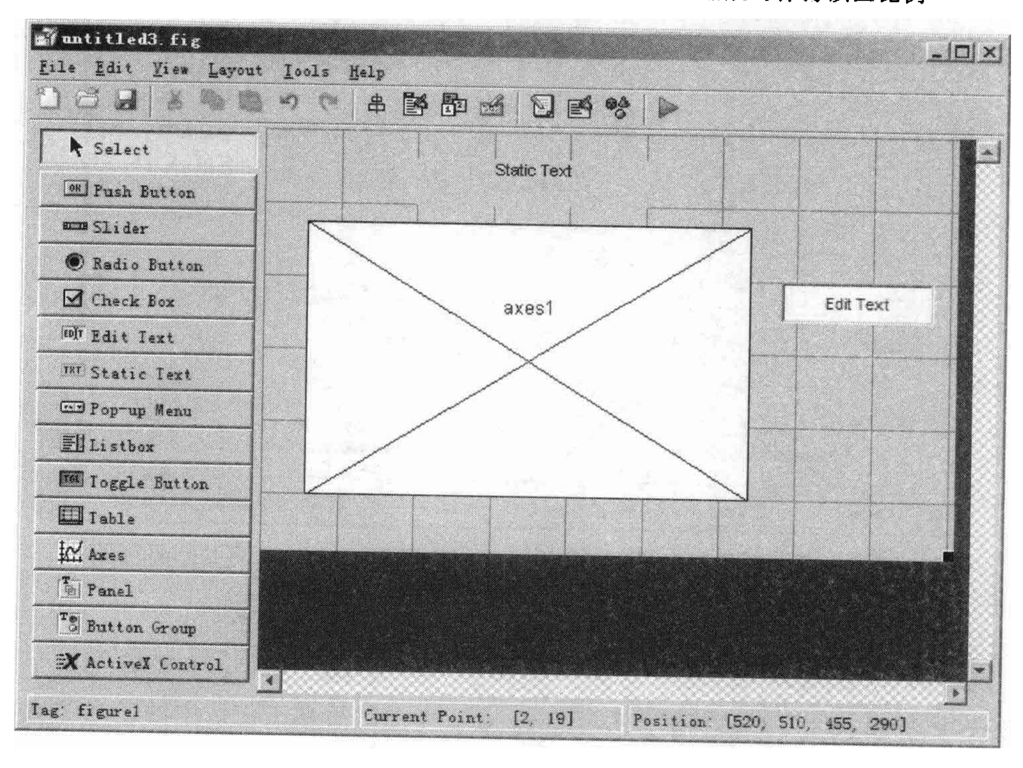


图 8.1-4 选配界面组件后的界面编辑器

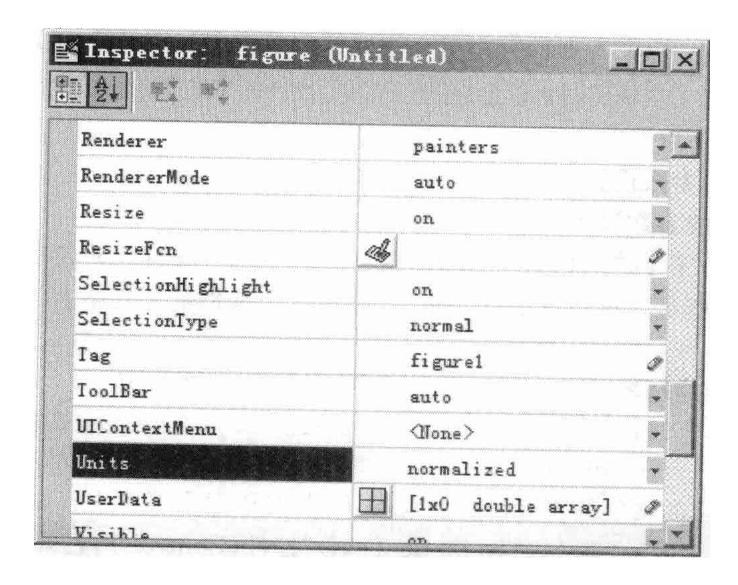


图 8.1-5 窗属性编辑器和设置的属性值

● 双击“轴(Axes)”组件(Components)，打开轴属性编辑器，设置下列属性值：

Box　　　on　　　　　　%轴框封闭
Units　　normalized　　%采用相对度量单位，缩放时保持比例
XLim　　[0, 15]　　　　%X 轴范围
　　　　　　　　　　　　%设置步骤：先点击 XLim 字条；然后填写数字
Ylim　　 [0, 2]　　　　%Y 轴范围

● 双击“可编辑文本(Edit Text)”控件，引出可编辑文本属性编辑器，作下列属性设置：

FontUnits　normalized　　%采用相对度量单位，缩放时保持字体比例
FontSize　　0.5　　　　　%字体大小
String　　　　　　　　　 %删去所有“缺省字符”，使编辑框呈现空白
Units　　　normalized　　%采用相对度量单位，缩放时保持该区比例

● 经以上属性设置后，设计界面呈现如图 8.1-6 所示的外观。

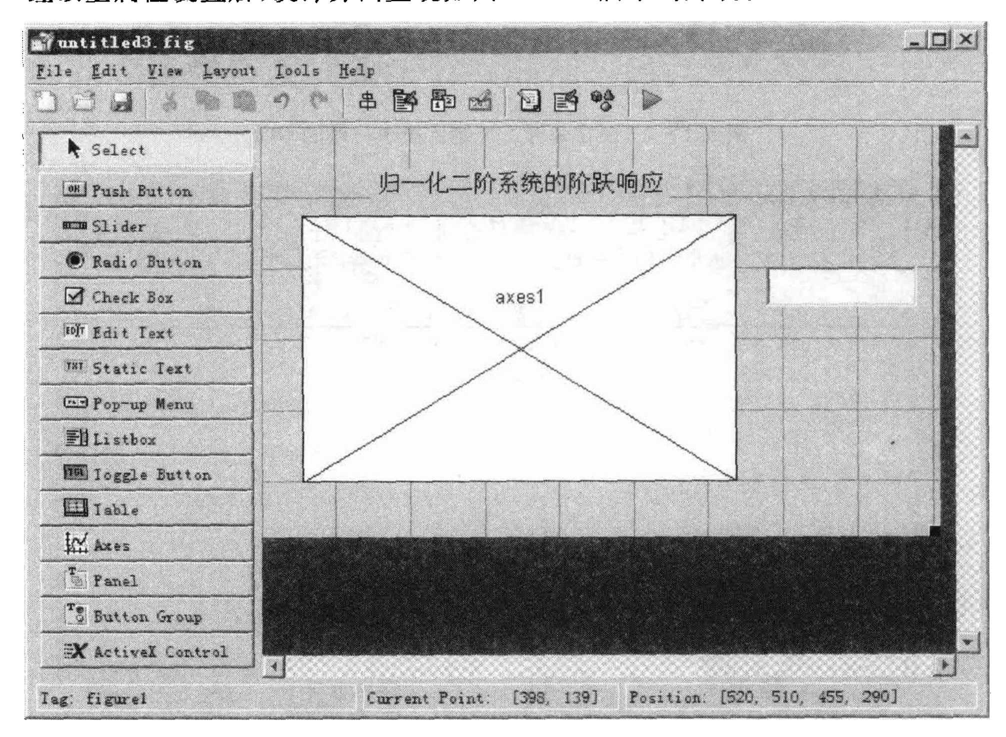


图 8.1-6　经属性设置后的设计界面

(5) 初步设计界面的保存

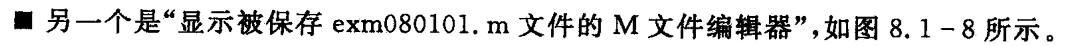
● 点击工具条上的▶图标，弹出“确认提示”对话框。

● 在该对话框中，点击[是(Y)]键，引出 Windows 系统的标准“保存”对话窗。

● 在“保存”对话窗中，选择文件夹、起文件名(如对于本例，文件名为 exm080101)，就完成了保存。在指定的文件夹上会同时生成 2 个伴生文件 exm080101.m 和 exm080101.fig。

● 随后，立即弹出两个窗口：

■ 一个是由 exm080101.fig 所呈现的与题目要求外形一致的“真实界面”，如图 8.1-7 所示。



■ 另一个是"显示被保存 exm080101. m 文件的 M 文件编辑器"，如图 8. 1 - 8 所示。

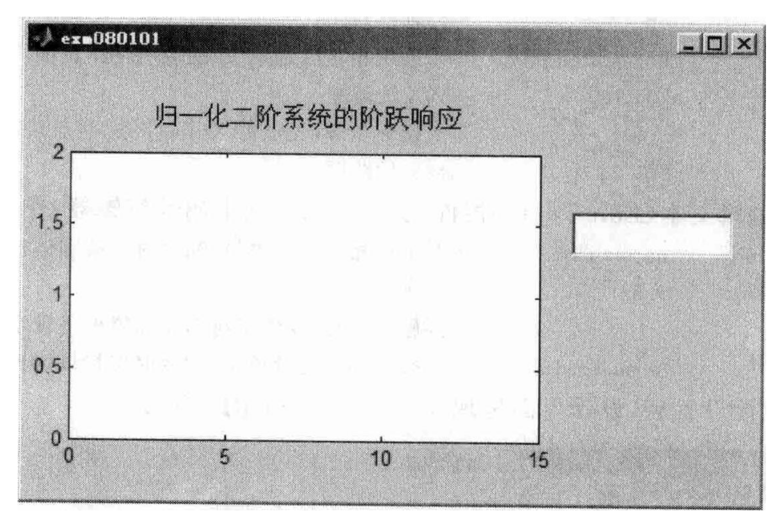


图 8. 1 - 7　仅在外形上与题给要求一致的 GUI

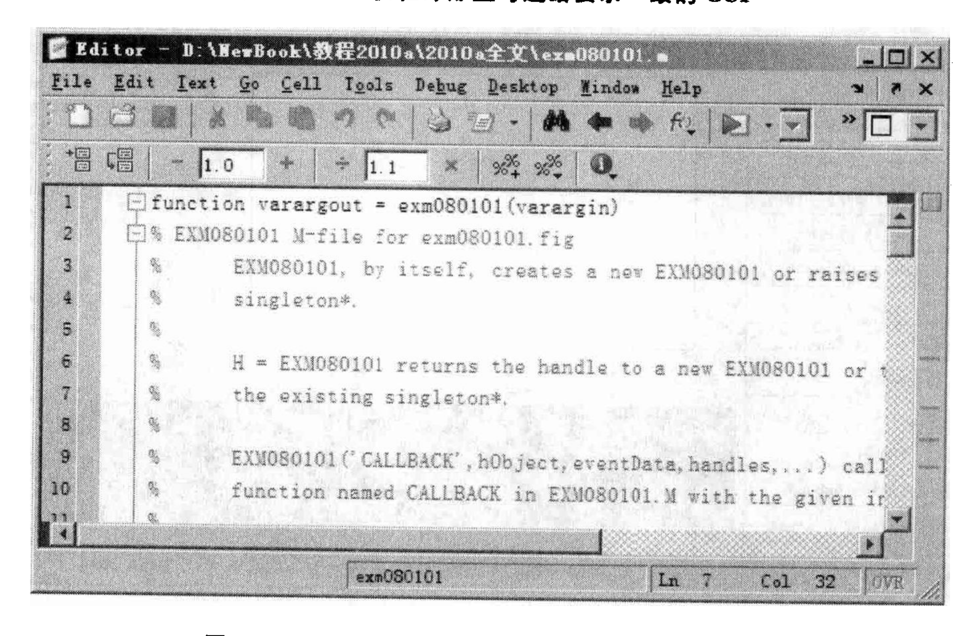


图 8. 1 - 8　界面编辑器自动生成的设计界面 M 文件

(6) 回调程序的编写

经以上步骤产生的界面，虽然在"可编辑文本框"中，可以输入阻尼比的数值，但还不具备"工作能力"。也就是说，它还不会根据输入的阻尼比，计算系统的阶跃响应，并在轴上绘制曲线。

为使界面具有"工作能力"，就必须在界面编辑器自动生成的 exm080101. m 程序的适当位置填写相应的 M 码指令。具体如下步骤如下：

● 在显示 exm080101. m 的 M 文件编辑器上，点击工具条中的 图标，在引出的菜单中选择回调子函数 edit1_Callback，如图 8. 1 - 9 所示。

● 仔细观察图 8.1－9，可以发现：目前，界面编辑器自动生成的 exm080101.m 所包含的 edit1_Callback 回调子函数的函数体是“空”的（见图中第 81 行处）。

● 要 edit1_Callback 回调子函数产生“动作”，就应在这“空白函数体中（即图 8.1－9 第 81 行处）”插入下列“工作指令”。

```
get(hObject,'String');                              %从编辑框读取输入字符
zeta=str2double(get(hObject,'String'));             %把字符转换成双精度数
handles.t=0:0.05:15;                                %定义时间采样数组
handles.y=step(tf(1,[1,2*zeta,1]),handles.t);       %计算响应
cla                                                 %清空坐标轴
line(handles.t,handles.y)                           %在已有轴上绘制曲线
```

● 图 8.1－10 所示即为编写了工作指令后的 edit1_Callback 回调子函数模样。

● 此后，再点击界面编辑器（或 M 文件编辑器）上的▶图标，此时弹出的界面就可接受输入数字，并绘图。

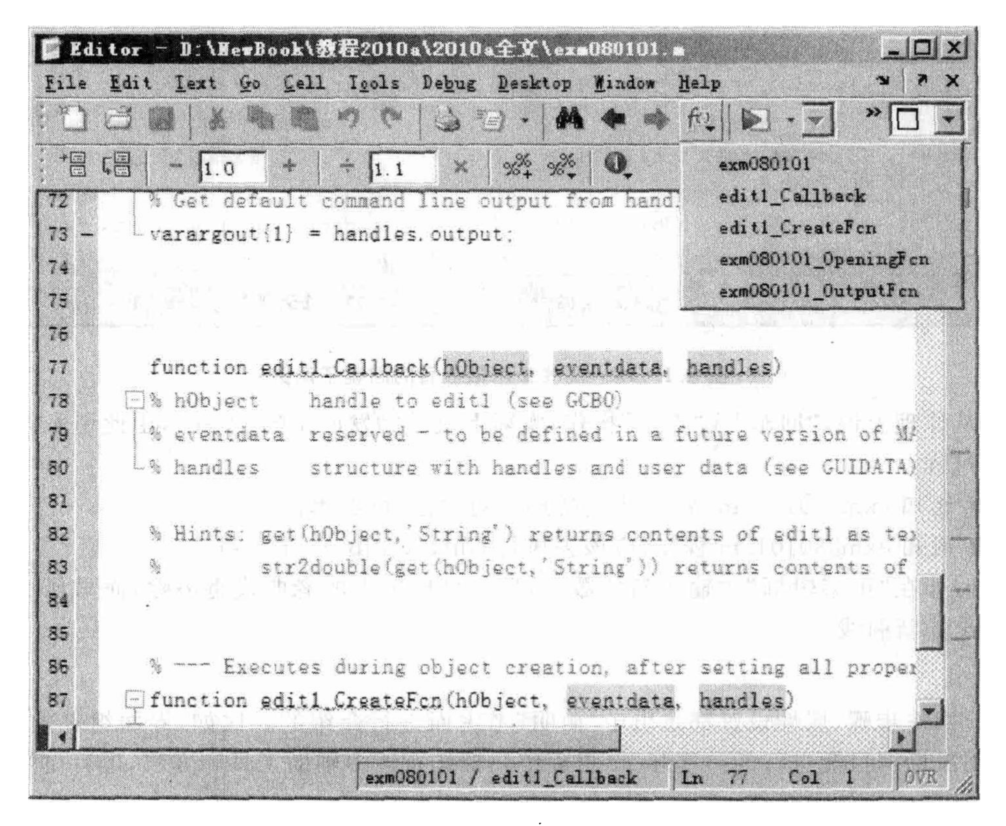


图 8.1－9　GUIDE 生成的“空白回调函数体”示图

(7) 所建界面的使用

● 在如图 8.1－7 所示的弹出界面的可编辑框中键入数字 0.3。注意：应在英文而非中文状态下输入。

● 按“回车”键，或在用户界面任何位置“点击”鼠标，就可在坐标轴上画出相应的曲线。参见图 8.1－1。

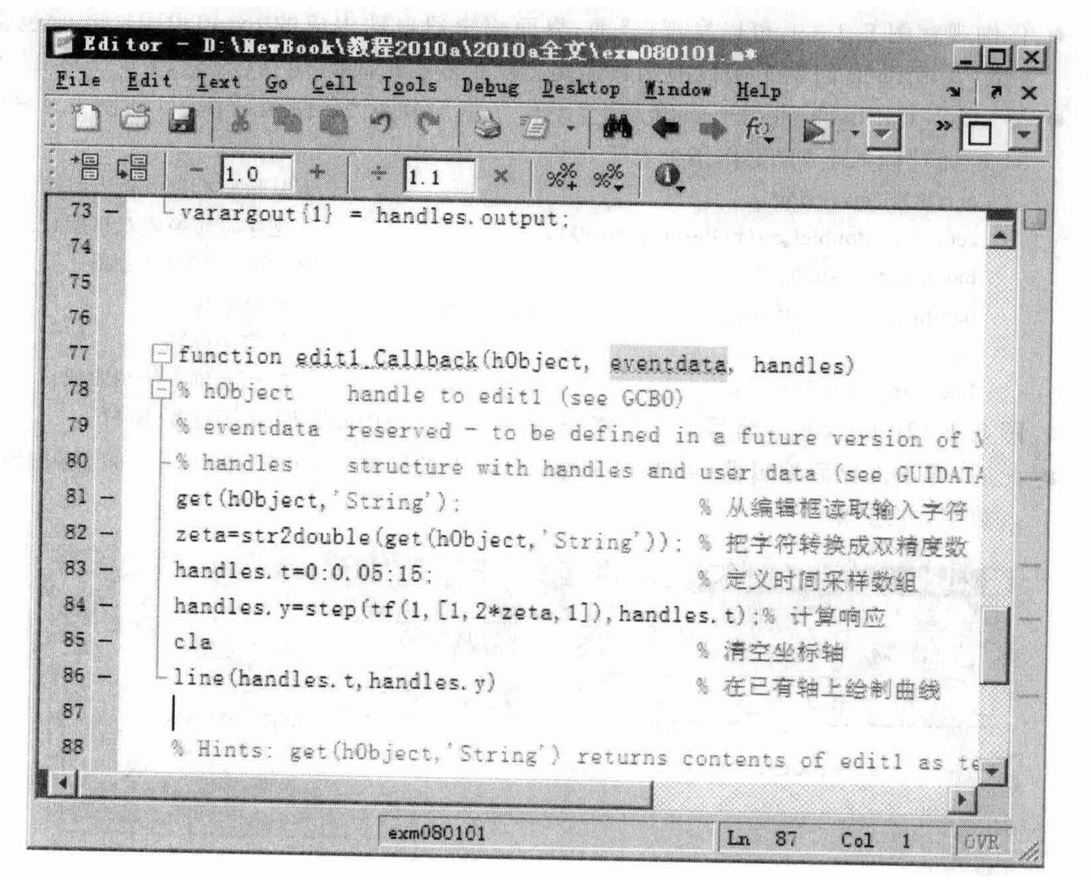


图 8.1-10　填写工作指令后的回调子函数

- 从原理上说，“回车”或“点击”操作，实际上就是触发回调的“事件”，由此引发以下两个过程：
  - 通知 exm080101.m 程序，当前的操作对象是“可编辑框”；
  - 通知 exm080101.m 程序，应该去执行 edit1_Callback 子函数。
- 假如在“可编辑框”中输入新的数字，那么坐标轴上的老曲线将擦除，而只显示重新画出的新曲线。

说明

以上操作步骤、属性设置都是出于“简明”考虑而进行介绍的。比如，界面组件的大小、位置都是用鼠标据目测进行的。实际上，界面编辑器还有许多辅助工具和方法，进行更精确地设置，将在下节的示例中介绍。

8.2　控件创建及应用示例

本节仍以算例为依托，详细讲述用户界面的创建步骤和注意要点。本节算例有两种好的学习方法：一种是循例而进，步步实践；另一种是参照步骤，独立实践。前者比较容易成功，后者更具挑战性、更培养能力。对于本节，最忌讳的学习方法是，只看文字，不动手操作。

本节算例是本章的核心算例。该算例所涉内容比较广泛，有的触及较深的 MATLAB 低层绘图指令，本书作者对此都给与简明的注释。假如读者能仔细阅读和耐心实践，那么定能对用户界面创建获得全方位的理解。

例【8.2-1】 为归一化二阶系统 $G(s)=\frac{1}{s^2+2\zeta s+1}$ 单位阶跃响应制作如图 8.2-1 所示的用户界面。

要求：①刚启动的界面初始形态如图 8.2-1 所示。②通过编辑框和滑键都能输入阻尼比。③在刚启动的初始界面上，响应曲线用红线绘制；而一旦界面被操作，则响应曲线将用蓝线绘制。④在列表框中的三个选项可以任意组合。

本例目的：①系统、完整地描述用户界面的创建步骤。②展示界面组件：轴、组件盘、可编辑文本框、滑键、列表框、无线电按键等控件的关键属性。③比较深入地了解 GUI 界面编辑器自动生成 M 函数文件的结构。④初始化子函数，以及滑键、无线电按键回调子函数的编写。⑤可编辑框的"标量数字输入"使用法下，回调子函数的编写。⑥列表框的"选项行序号"使用法下，回调子函数的编写。⑦"GUI 数据(GUI Data)"在各子函数间的传递和共享。

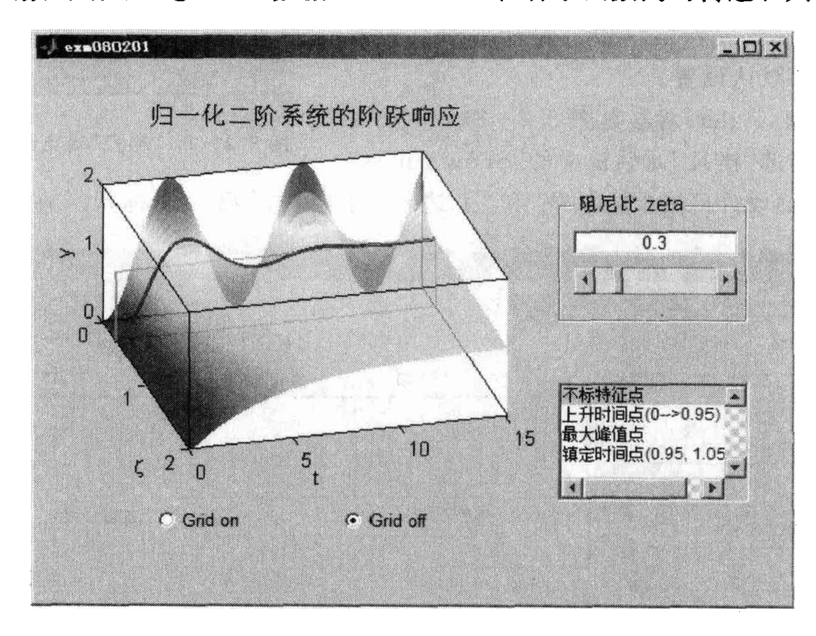


图 8.2-1　用户界面的初始状态

(1) 开启 GUI 界面编辑器

- 在 MATLAB 指令窗中运行 guide 指令，引出"GUIDE Quick start"对话窗(参见图 8.1-2)。
- 在"新建 GUI(Create New GUI)"页面，点击[OK]键，引出空白的界面设计环境(参见图 8.1-3)。

(2) 界面窗属性设置

- 拖拉"界面设计工作区"右下角的"小黑方快"句柄，使其长宽调整到希望的大小，即未来应用界面出现时的"默认大小"。
- 双击界面编辑器的界面设计工作区，引出"窗属性编辑器(Inspector:figure)"(参见图

8.1-5)。在“窗属性编辑器”中，设置下列属性值：

Resize on %该设置很重要。它决定图形窗是否可缩放。

注意：这种属性设置，总可以保证未来界面窗的“可缩放性”，而不管此前是否利用“GUI Options”对话窗进行过设置。

(3) 为界面设计区引入坐标参照系

● 坐标参照系的一般性介绍

为便于在界面上进行控件的布置，MATLAB 提供了如下常用的参照工具：纵横分格线、标尺、基准线和邻线对准功能。引入参照系的操作过程如下：

■ 点选 GUI 界面编辑器上的菜单项{Tools > Grid and Rulers}，引出如图 8.2-2 所示的“格尺”对话框。

■ 勾选“格尺”对话框中相应选项，就可以得到相应的参照工具和功能。

■ 图 8.2-2 展示了“格尺”对话框对纵横方格线的默认设置。

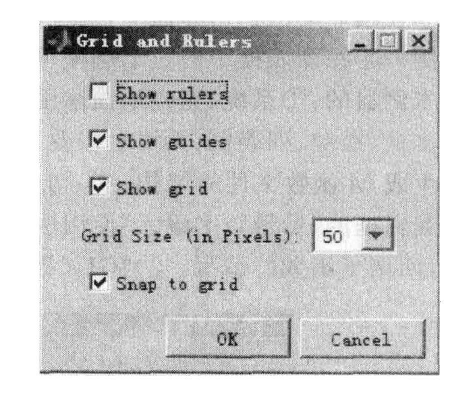


图 8.2-2 “格尺”对话框(默认状态)

● 纵横标尺(Rulers)(参见图 8.2-3)

■ 只有勾选“格尺”对话框中的“Show rulers”项，界面设计区的“左边界”和“上边界”才会显示出纵横标尺。在本例中，“Show

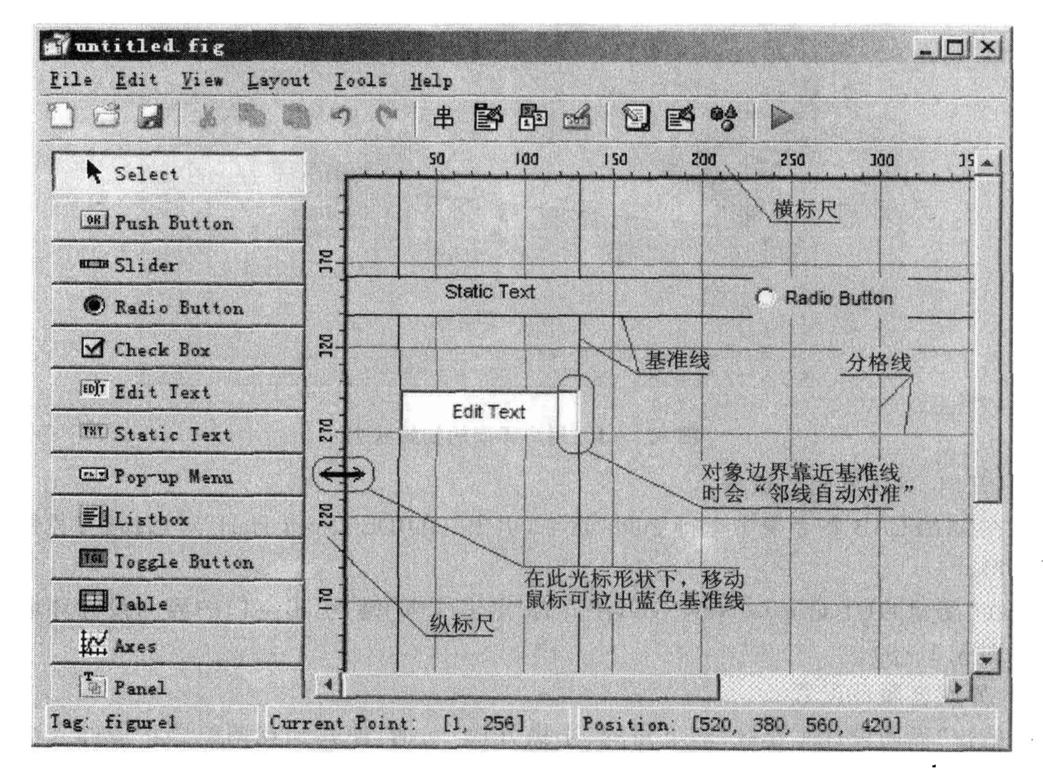


图 8.2-3 一般坐标参照工具示意图

rulers"项已被勾选。

■ 不管界面设计区(即未来界面窗对象)的"Units"属性如何设置,标尺单位总是为象素。设计区左下角的标尺显示是(0, 0)。

● 纵横方格线(Grid)(参见图 8.2-3)

■ 只有勾选"格尺"对话框中的"Show grid"项,设计区中才出现纵横方格线。

■ 方格线的间距可以通过"Grid Size"在 10~200 像素之间选定。缺省间距为 50 像素。

● 纵横基准线(Guide Lines)

■ 基准线产生的前提是纵横标尺的存在。

■ 在"Show rulers"项勾选的前提下,再勾选"Show guides"项;然后,把鼠标移到左(或上)标尺上,光标形状由"空心箭头"变为"双向箭头"时,按住鼠标,往右(或下)拉动,就能引出"蓝色的基准线"。(参见图 8.2-3)

■ 基准线可设置任意多条,也可用鼠标任意拖动。

● 邻线对准功能(Snap-to-Grid)(参见图 8.2-3)

■ 勾选"Snap to grid"项,或者勾选 GUI 界面编辑器菜单项{Layout > Snap to Grid},都可启动该功能。

■ 在启用该功能后,当鼠标拖动的界面设计区内任何组件的某一"边界"靠近"方格线"或"基准线"到一定程度时,该组件会自动使该"边界"与"邻近的方格(基准)线"对准。

■ 该功能的启用可提高鼠标排版的精准程度。

● 注意:图 8.2-4 显示了本例所采用的基准线。

(4) 根据题目要求进行界面构建

由于本例题目所给界面明确,所以可以直接进行组件的布置。

● "轴"组件的配置

■ 在"组件模板区",点选"轴(Axes)"图标 Axes 。

■ 在设计工作区的适当位置,用鼠标拉出适当大小的"轴位框"(参见图 8.2-4),以供绘制响应曲线使用。

■ 双击轴位框,在引出的"属性编辑器(Property Inspector)"中,进行如下设置:

FontSize	0.065	
FontUnits	normalized	%采用相对度量单位,缩放时保持字体比例
Units	normalized	%缩放时保持轴与界面之间的比例
XLimMode	auto	%使适应三维图形
YLimMode	auto	%使适应三维图形

● "静态文本框"组件的配置

■ 点击模块区"静态文本框(Static Text)"图标 Static Text ;通过鼠标的拖拉操作,把该文本框的设置在"轴位框"的上方;文本框的大小也可以用鼠标调节(参见图 8.2-4)。

■ 双击静态文本框,在引出的"属性编辑器(Property Inspector)"中,进行如下设置:

FontSize	0.5	%框内字体大小(框高度为 1)
FontUnits	normalized	%采用相对度量单位,缩放时保持字体比例
String	归一化二阶系统的阶跃响应	

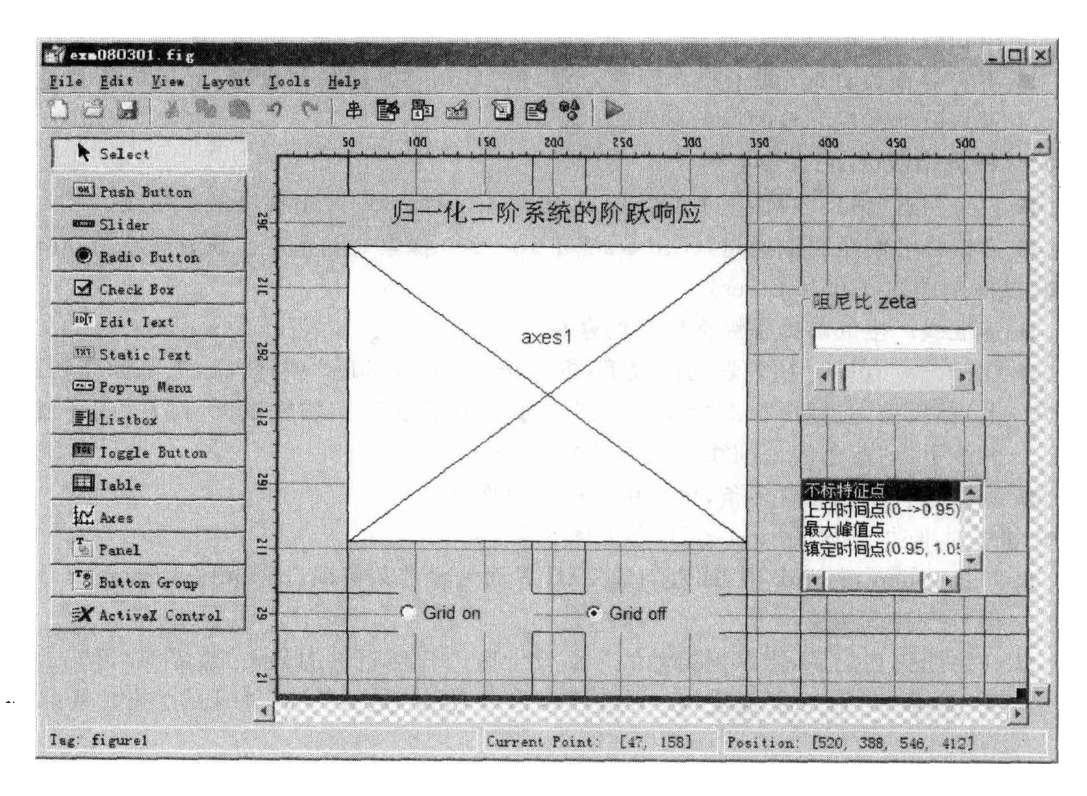


图 8.2-4　设计本例用户界面的界面编辑器

Units	normalized	%缩放时保持轴与界面之间的比例

● “组件盘”的引入

■ 点击模块区“组件盘(Panel)”图标 Panel ；在“轴位框”右侧拉出足够大的区域，以容纳“可编辑框”和“滑键”(参见图 8.2-4)。

■ 双击该“组件盘”，在引出的“属性编辑器”中，进行如下设置：

FontSize	11	
FontUnits	points	%先选此单位，待 Fonsize 选定后，再选为 normalized。
Title	阻尼比 zeta	%组件盘名称
TitlePosition	fifttop	%组件盘名称的位置
Units	normalized	

在此顺便指出：当字体大小单位(FontUnits)选为 normalized 时，FontSize 总显示为 −1，而无法设置。

● “可编辑框”控件的配置

■ 点选“可编辑文本框 (Edit Text)”图标 Edit Text 。

■ 用鼠标在组件盘内的适当位置，拉出大小合适的编辑显示区(参见图 8.2-4)。注意：“先有组件盘存在，然后把其他组件往组件盘中放”的次序不能颠倒。否则，当用鼠标移动组件盘时，那些似乎在盘中的组件是不会随之移动的。

■ 双击可编辑框，在引出的“属性编辑器”中，进行如下设置：

FontSize	0.7	%框内字体大小(框高度为 1)

FontUnits	normalized	%采用相对度量单位,缩放时保持字体比例
String		%初始显示(空白)
Units	normalized	%缩放时保持轴与界面之间的比例

● “滑键”的配置

■ 点选“滑键(Slider)”图标 Slider 。

■ 在组件盘区域内,拉出适当大小的滑键(参见图 8.2-4)。

■ 双击该滑键,在引出的“属性编辑器”中,进行如下设置:

FontSize	0.5	%框内字体大小(框高度为 1)
FontUnits	normalized	%采用相对度量单位,缩放时保持字体比例
Max	2	%滑键定义阻尼比的最大值
Min	0	%滑键定义阻尼比的最小值
SliderStep		
x	0.01	%箭端操纵下,游标的滑动步长
y	0.1	%游标直接移动时的滑动步长
Units	normalized	%采用相对度量单位,缩放时保持该键比例
Value	0	%使游标在最左端

● “无线电按键”的配置

因为要配置两个无线电按键,故此以下操作要进行两次。

■ 点选“无线电按键(Radio Button)”图标 Radio Button 。

■ 在轴位框的下方,拉出适当大小的无线电按键(参见图 8.2-4)。

■ 双击该键,在“属性编辑器”中,进行如下设置:

FontSize	0.7	
FontUnits	normalized	
String	Grid on	%按键在界面上呈现的名称
Tag	gridon	%使该键回调子函数名为 gridon_Callback
Units	normalized	
Value	0	%此值对应“非选”标识态

■ 另一个无线电按键只有两个属性的设置与前一个不同,具体如下:

String	Grid off	
Tag	gridoff	%使该键回调子函数名为 gridoff_Callback
Value	1	%此值对应“选中”标识态

● “列表框”的配置

■ 把“列表框(Listbox)”模块 Listbox 拉到合适位置,并调节大小(参见图 8.2-4)。

■ 双击该列表框,在引出的“属性编辑器”中,进行如下设置:

FontSize	0.17	
FontUnits	normalized	
Max	2	%只有当 Max－Min>1 时,才点选允许“多 %个选项”。
Min	0	
String	不标特征点	
	上升时间点(0→0.95)	

	最大峰值点	
	镇定时间点(0.95, 1.05)	
Units	normalized	
Value	1	%使界面上列表框中第 1 行选中

● 点击界面编辑器工具条上的▶图标，引出如图 8.2-5 所示的图形窗(它的相应文件为 exm080201.fig)和 M 文件编辑器(在编辑器中显示执行文件 exm080201.m)。

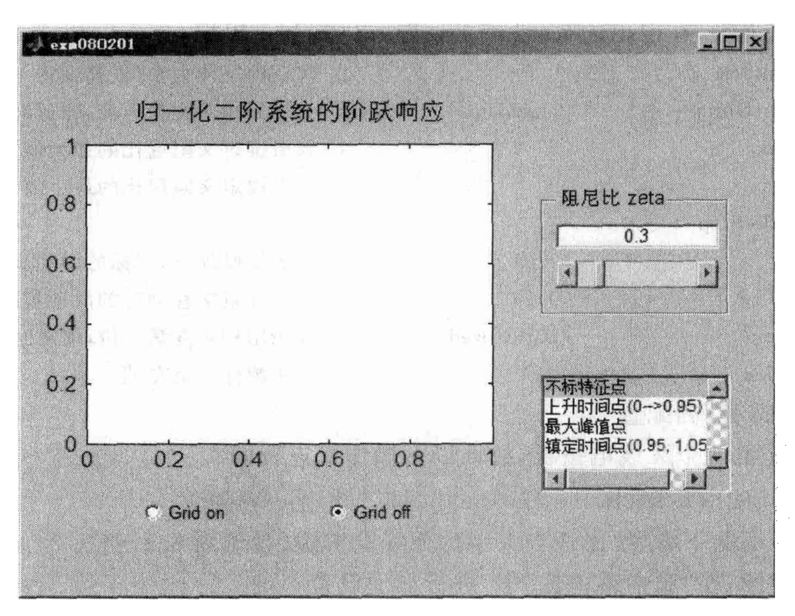


图 8.2-5　执行文件尚未填写时的生成界面

(5) 由界面编辑器自动产生的 exm080201.m 文件的结构

该 M 文件为一个主函数，内含 11 个子函数，具体结构及相关说明如下。

● 主函数

```
function  varargout = exm080201(varargin)
```

● 子函数

■ 界面启动子函数和输出子函数

```
function exm080201_OpeningFcn(hObject, eventdata, handles, varargin)
function varargout = exm080201_OutputFcn(hObject, eventdata, handles)
```

■ 编辑框回调子函数组

```
function  edit1_Callback(hObject, eventdata, handles)
function edit1_CreateFcn(hObject, eventdata, handles)
```

■ 滑键回调子函数组

```
function slider1_Callback(hObject, eventdata, handles)
function slider1_CreateFcn(hObject, eventdata, handles)
```

■ 无线电按键 Grid on 回调子函数

```
function gridon_Callback(hObject, eventdata, handles)
```

■ 无线电按键 Grid off 回调子函数

```
function gridoff_Callback(hObject, eventdata, handles)
```

■ 列表框回调子函数组

```
function listbox1 Callback(hObject, eventdata, handles)
function listbox1 CreateFcn(hObject, eventdata, handles)
function listbox1 KeyPressFcn(hObject, eventdata, handles)
```

● (应该指出)上述函数结构具有"典型性":

■ 每个由界面编辑器产生的 M 文件都包含主函数、界面启动子函数和输出子函数。而且,子函数名的构成形式也都一样,即由"制作者输入的保存文件名"加固定的 OpeningFcn或 OutputFcn 字节构成。

■ 回调子函数组与用户界面上的控件相对应。比如在本例中,界面上有 5 个控件,就对应 5 个回调子函数组。这 5 组子函数名的"头字节"取自 5 个控件的识别名属性 Tag 的取值:"edit1"、"slider1"、"gridon"、"gridoff"、"listbox1"。

■ 在上述子函数中,只有"字体加粗"的子函数需要界面制作者填写 M 码,以实现该界面控件被触发后,产生相应操作,实现目标。

(6) 编写初始化程序

由界面编辑器独立生成的 exm080201_OpeningFcn 启动子函数一般是不完整的,而必须由界面制作者根据任务填写相应的 M 码。

启动子函数的执行时间发生在 GUI 所有组件建立以后(即 CreateFcn 运行之后),但在将它们显示在屏幕上之前。启动子函数的一般任务是:为 GUI 的使用准备数据和界面形态。本例启动子函数 M 码执行的任务是:

● 初始化设定,包括对界面是否初始态加设标志,使编辑框显示初始阻尼比,使滑键游标位置反映阻尼比,使列表框显示初始选项,使坐标不带分格线。

● 准备以后绘图所需的时间采样数组。

● 为初始界面绘制曲线。

● 以上准备工作完成后,把初始化数据借助"GUI 数据"形式保存在变量 handles 中。

请读者阅读下列 exm080201_OpeningFcn 子函数时注意:该子函数已经过本书作者完善。整个启动子函数分为两部分:

● 处于"%U_Start"与"%U_End"之间的 M 码,以及所有中文注释都是本书作者添加的。

● 其余 M 码都是由 GUIDE 平台自动生成的,通常是不完整的。

```
function exm080201_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to exm080201 (see VARARGIN)
%U Start ---"%U Stat"和"%U End"之间的 M 码均由作者编写-----U Start
zeta = 0.3;                                  %初始阻尼比
set(handles.edit1,'String',num2str(zeta))    %编辑框标识初始化,注意:0.3 变为字符串
set(handles.slider1,'Value',zeta)            %滑键游标位置的初始化
set(handles.gridon,'Value',0)                %Grid on 无线电按键处于"非选"状态
```

```
set(handles.gridoff,'Value',1)           % Grid off 无线电按键处于"选中"状态
set(handles.listbox1,'Value',1)          % 列表框选项初始化,用第 1 选项
handles.t = 0:0.05:15;                   % 定义时间采样数组                    <14>
handles.Color = 'Red';                   % 定义响应曲线的初始色彩              <15>
handles.zeta = zeta;                     %                                     <16>
handles.flag = 0;                        % 初始绘图标志
handles = surfplot(handles);             %                                     <18>
handles.flag = 1;                        % 非初始绘图标志                      <19>
handles.Color = 'Blue';                  % 定义响应曲线的非初始色彩            <20>
% U End ------------------------------------------------------------------U End
handles.output = hObject;                %                                     <21>
guidata(hObject, handles);               % 此前指令改变了 handles,必须靠 guidata 指令才能
                                         % 把更新了的 handles 加以保存,以供后用。
```

(7) 可编辑框的回调子函数

本例回调子函数执行任务的过程如下:

- 界面上对编辑框的操作,使 gcbo 得到该框句柄,并把此句柄用作 edit1_Callback 函数的第 1 个输入量 hObject。
- 获取界面控件状态:列表框的选项;编辑框的"String 属性值"。该属性值是"字符串"数据类型的。因此,如需作为"数值"使用,就必须经过转换(见指令<7>)。
- 使界面上其他控件(滑键游标)状态与"新阻尼比"对应(见指令<8>)。
- 在新阻尼比下,计算阶跃响应,重新绘图。

再次提醒:以下 edit1_Callback 子函数中,"%U_Start"与"%U_End"间的 M 码,以及所有中文注释都是本书作者添加的。其余均由界面编辑器自动生成。(本章其它地方,类似标识的含义相同,不再说明。)

```
functionedit1_Callback(hObject, eventdata, handles)
% hObject    正被操作的控件,即可编辑的句柄
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    保存界面所有组件句柄和用户数据的构架变量,即 GUI 数据变量
% U Start --------------------------------------------------U Start
sz = get(hObject,'String');              % 从编辑框读取输入字符              <6>
zeta = str2double(sz);                   % 把字符转换成双精度数              <7>
set(handles.slider1,'Value',zeta)        % 对滑键的游标定位
handles.zeta = zeta;                     % "GUI 数据"形式,保存数据以便共享  <9>
handles = surfplot(handles);             % 调用绘图子函数,绘制响应曲线
guidata(hObject, handles);               % 保存更新了的 handles 数据         <11>
% U End ----------------------------------------------------U End
```

(8) 滑键回调子函数

在编写该子函数时,要考虑的是:游标值的读取;使可编辑框显示此值;更新 handles,重新绘线。为叙述简洁,该子函数的执行任务见下列 M 码中的中文注释。

```
function slider1 Callback(hObject, eventdata, handles)
%U Start-------------------------------------------------U Start
zeta = get(hObject,'Value');                        % 读取游标体现的"值"          <3>
set(handles.edit1,'String',num2str(zeta))           % 使编辑框显示新阻尼比        <4>
handles.zeta = zeta;                                % 更新阻尼比                  <5>
handles = surfplot(handles);                        % 利用 handles 传送绘图所需数据,  <6>
%并产生新的 handles 数据。
guidata(hObject, handles);                          %                             <7>
%U End---------------------------------------------------U End
```

(9) 无线电按键回调子函数

每个无线电按键各对应一个回调子函数。本例中两个按键的任务是:一,处理坐标的分格线;二,两个按键的相互制约。为叙述简洁,该子函数的执行任务见下列 M 码中的中文注释。

```
function gridon Callback(hObject, eventdata, handles)
%U Start-------------------------------------------------U Start
set(handles.gridoff,'Value',0)                      % 使"Grid off"键处于"非选"状态
grid on                                             % 画坐标分格线
%U End---------------------------------------------------U End

function gridoff Callback(hObject, eventdata, handles)
%U Start-------------------------------------------------U Start
set(handles.gridon,'Value',0)                       % 使"Grid on"键处于"非选"状态
grid off                                            % 删除坐标分格线
%U End---------------------------------------------------U End
```

(10) 列表框回调子函数

为叙述简洁,该子函数的执行任务见下列 M 码中的中文注释。

```
function listbox1 Callback(hObject, eventdata, handles)
%U Start-------------------------------------------------U Start
listindex = get(hObject,'Value');                   % 获得列表框所有选项的序号
                                                                                  <3>
if any(listindex == 1)                              % 使第 1 选项只能单独被选
    set(handles.listbox1,'Value',1)
end
handles.flag = 0;                                   % 列表框被触动时,发出重画曲面指令  <7>
handles = surfplot(handles);
handles.flag = 1;                                   %                             <9>
guidata(hObject, handles);                          %                             <10>
%U End---------------------------------------------------U End
```

(11) 绘图子函数

由于在启动子函数及三个控件回调子函数中都需要几乎相同的绘制曲线的程序,所以本书作者把它们设计成一个独立的子函数,供启动函数和各回调子函数使用。

```
function handles = surfplot(handles)
% handles = surfplot(handles)     供启动子函数和各控件回调子函数调用的绘图函数
```

```
% handles                    GUI 数据变量。该变量保存和传递:界面上各种图形对象的
%                            "句柄",以及"应用参数"。
zeta = handles.zeta;                                    % 仅为记述简单
t = handles.t;
listindex = get(handles.listbox1,'Value');              % 读取列表框的选项序号数组
Nt = length(t);
if handles.flag = = 0                                   % 仅在界面启动时执行
    cla                                                 % 清空界面上次工作后可能残留的轴上对象
    zmin = get(handles.slider1,'Min');                  % 读取滑键的最小取值
    zmax = get(handles.slider1,'Max');                  % 读取滑键的最大取值
    zt = zmin:0.05:zmax;                                % 为三维坐标"x 轴"准备采样点
    Nz = length(zt);
    [ZT,T] = meshgrid(zt,t);                            % 三维曲面的"X"、"Y"采样点阵
    Y = zeros(Nt,Nz);
    for k = 1:Nz                                        % 计算不同阻尼下的响应曲线
        Y(:,k) = step(tf(1,[1,2 * zt(k),1]),t);
    end
    surface(ZT,T,Y)                                     % 用低层绘图指令,绘制曲面
    shading flat
else
    delete(handles.gl)                                  % 利用句柄,删除上次绘制的参照线
    delete(handles.rline)                               % 利用句柄,删除上次绘制的响应曲线
end
xz = ones(1,Nt) * zeta;
y1 = ones(1,Nt) * 1;
y = step(tf(1,[1,2 * zeta,1]),t);                       % 在指定的 zeta 下,计算响应曲线
gz = [zeta,zeta,xz,zeta,zeta,xz];                       % 绿色封闭参照线的"x"坐标
gt = [t(1),t(1),t,t(end),t(end),fliplr(t)];             % 绿色封闭参照线的"y"坐标
gy = [0,1,y1,1,0,0 * y1];                               % 绿色封闭参照线的"z"坐标
handles.gl = line(gz,gt,gy,'Color','g','LineWidth',1);  % 绘制绿色参照线,并产生句柄      <33>
handles.rline = line(xz,t,y,'Color',handles.Color,'LineWidth',2); % 绘制红色响应曲线,并产生句柄
K = length(get(handles.listbox1,'Value'));              % 列表框里,被"选中"的项数
for jj = 1:K
    switch listindex(jj)                                % 被选中的"选项行序号"            <37>
        case 1
                                                        % 不做任何标识
        case 2                                          % 画上升时间点
            k95 = min(find(y>0.95));k952 = [(k95 - 1),k95];
            t95 = interp1(y(k952),t(k952),0.95);        % 线性插值法确定 0.95 线的时间
            line(zeta,t95,0.95,'marker',' + ','markeredgecolor','k','markersize',6);
        case 3                                          % 画最大峰值点
            [ym,km] = max(y);                           % 找最大峰值
            ifkm<Nt & (ym - 1)>0
```

```
                line(zeta,t(km),ym,'marker','.','markeredgecolor','k','markersize',5);
            end
        case4                                                   % 画镇定时间点
            ii = max(find(abs(y - 1)>0.05));
            if ii<Nt
                line(zeta,t(ii + 1),y(ii + 1),'Color','r','Marker','o','MarkerSize',5)
            end
    end
end
xlabel('{\zeta}')
ylabel('t')
zlabel('y')
alpha(0.7)                                                      % 控制曲面的透明度
view(75,44)
```

(12) 用户界面的运行

- 用户界面的文件
  - ■ 由界面编辑器制成的用户界面分存为两个伴生文件。本例界面的伴生文件是 exm080201.m 和 exm080201.fig。
  - ■ 在用界面编辑器开启用户界面时,所选择的是 FIG 文件。在界面编辑器设计区里显示的是界面的组件几何图形。若要显示相应的 M 文件,须点击图标,引出 M 文件编辑器。
- 用户界面的导出
  - ■ 制作完成的用户界面是不依赖界面编辑器的。
  - ■ 只要用户界面的伴生 FIG 文件和 M 文件在同一目录上,且这个目录是 MATLAB 的当前目录,或该目录在 MATLAB 的搜索路径上,那么运行 M 文件就可导出用户界面。
  - ■ 对于本例,在 M 文件编制完成后,或在 M 文件编辑器中,或在界面编辑器上,点击工具条中的▶图标,就能呈现如图 8.2-1 所示的初始界面。
- 在界面列表框中点选全部特征点选项,并使滑键游标连续步进,就会在曲面上画出三条特征线:"小红圈线"表现在不同阻尼比下的上升时间;"小黑点线"表示最大峰值点随阻尼比的变化;"黑十字叉线"表现上升时间随阻尼比的变化,如图 8.2-6 所示。
- 本界面大小可以任意缩放;彩色曲面反映二阶系统响应曲线与阻尼比的关系;坐标轴分格线有无可以控制;阻尼比既可以从编辑框输入,也可以借助滑键使阻尼比"连续步进"变化,观察响应曲线的连续变化。

说明

- 借助界面编辑器及其辅助设计工具,通过鼠标操作进行用户界面配置,不仅简化设计过程,而且所得的执行 M 文件格式规范。换句话说,借助界面编辑器制作用户界面,既方便快捷,而且所得界面文件也便于理解和维护。
- 在对界面编辑器生成的 M 文件填写 M 码之前,一定要仔细阅读那文件中的"原有注释"。凡写明"不得改动"的 M 码,制作者千万不要去改动,除非已经透彻理解 GUI 的

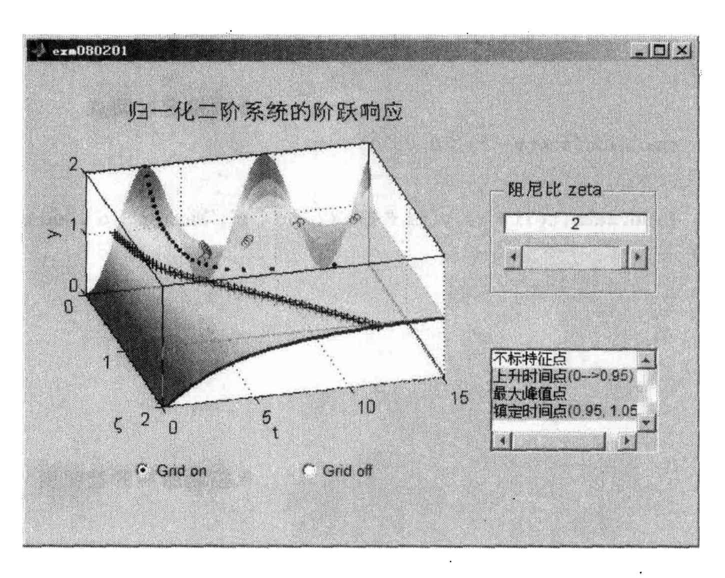


图 8.2-6　显示列表框全部选项的界面示例

工作机理。

■ 不要企图"一口气"完成界面执行 M 文件的编写，而应采用"边写边试边改"的方法逐步完善。

■ 子函数可以写一个，试验一个，不必过多顾虑各控件的协调，可以在试验中不断协调。

■ 在所有功能基本实现后，要对制成的用户界面从多个角度、不同运作次序加以检查，发现问题，逐步完善。切勿急于求成！

■ 在功能验证正确后，再对界面执行 M 文件的指令进行优化。

- 本例只演示了可编辑框的"标量数字输入"使用法。在只允许输入"标量数字"的情况下，从可编辑框"String"属性读得的是字符串。回调子函数 edit1_Callback 中的第〈7〉〈8〉行指令的功能就是：把字符串读出，再把字符串转换为"双精度"数。实际上，可编辑框也允许"多行字符输入"。限于篇幅，本教程不予阐述。有兴趣的读者可参见文献[1]或本书作者的其他著作，或利用 Edit Text 为关键词从 MATLAB 的 Help 浏览器获得帮助信息。
- 本例演示列表框的"选项行序号"使用法。回调子函数 listbox1_Callback 中的第＜3＞行指令和绘图子函数中从第＜37＞行指令开始的 switch-case 结构，是该使用法的具体表现。
- 在本例中，各子函数间的参数传递是通过"构架变量"handles 运用所谓"GUI 数据存取机制"实现的。

8.3　菜单及工具图标的设计示例

在 Windows 系统中，界面上通常都配置有菜单条（Menu Bar）和工具条（Tool Bar）。MATLAB 制作的 GUI 也同样可以很方便地配置菜单和工具图标。

本节将通过分置于两小节中的算例介绍以下内容：

- MATLAB 图形窗标准菜单和标准工具条的创建。
- 用户菜单的定制和标准图标的选用。

8.3.1 为界面配置标准菜单条和工具条

例【8.3-1】 在例 8.2-1 产生的界面上，配置 MATLAB 标准图形窗菜单。本例演示：如何利用已有界面制作新界面；通过对界面窗“MenuBar”和“ToolBar”属性的设置，产生标准菜单条和工具条。

(1) 以例 8.2-1 的用户界面为基础制作新界面

- 在 MATLAB 指令窗中运行 guide 指令，引出“GUIDE Quick start”对话窗。
- 在此对话窗“Open Existing GUI”页面上，选中 exm080201.fig 文件，再点击[Open]键，引出显示 exm080201.fig 界面的界面编辑器。
- 在此界面编辑器上，选中{File > Save As}菜单项，把它保存为 exm080301。
- 特别提醒：读者在 M 文件编辑器中，借助对 exm080201.m 实施“另存为”操作而获得 exm080301.m 的 M 文件，不可能引出所需的用户界面。

(2) 利用属性编辑器为界面配置标准菜单条和工具条

- 双击界面编辑器设计工作区的空白处，引出显示“界面窗”的属性编辑器。
- 在属性编辑器中，进行以下属性取值

```
MenuBar        figure        %采用图形窗的标准菜单条
ToolBar        figure        %采用图形窗的标准工具条
```

- 完成以上设置后，点击界面编辑器工具条上的▶图标，就可得到符合题目要求的如图图 8.3-1 所示的用户界面。

(3) 界面操作示范

- 在 exm080301 的图形用户界面上，用鼠标按住滑键上的滑动条，使之移动到滑键的最左端。在此操作过程中，界面上的蓝色响应曲线会随之变化。
- 用鼠标点选列表框中的“最大峰值点”菜单项。
- 用鼠标连续点击滑键的右端箭头键，使滑动条不断向右移动。在此操作的同时，可以看到界面上的蓝色响应曲线不断变化，且响应曲线的最大峰值点也用“小黑点”标出。
- 点选 exm080301 图形用户界面上的图标，或菜单项{Tools > Data Cursor}，光标变成“双十字”，再用它点击响应曲线的峰值点，就出现一个矩形显示框，该框显示该峰值点的三个数据。特别提醒注意：显示框中三个数据{x, y, z}分别对应三维坐标轴{ζ, t, y}，参见图 8.3-1。
- 经以上操作后的 exm080301 界面，参见图 8.3-1。

说明

- 本例演示了：如何利用已有用户界面，创建新界面。在此再次提醒：对于由 GUIDE 界面编辑器产生的用户界面，为制作新界面所进行的“另存为”操作，必须在界面编辑器平台上进行，而绝不能在 M 文件编辑器中进行。

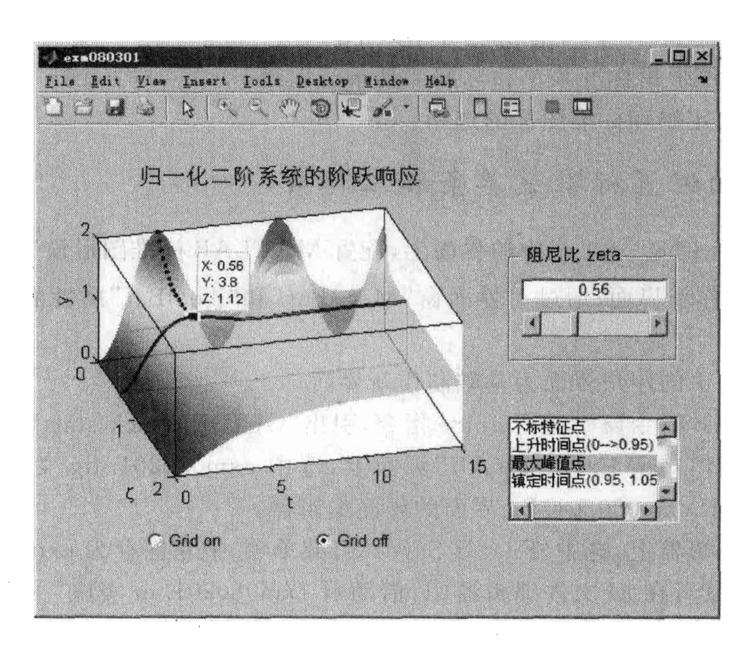


图 8.3-1　显示标准菜单和工具条的图形用户界面

- 在 GUIDE 界面编辑器中，开启的空白界面窗的“MenuBar”属性缺省设置为“none”，而“ToolBar”属性缺省设置为“auto”。因此，若不对这两个属性进行专门的设置，则界面编辑器设计出来的界面窗就不显示菜单条和工具条。
- 在用户界面上“增设标准菜单和工具条”的操作都非常简单，所产生的菜单和图标功能众多。因此，在既需要又许可的情况下，推荐使用标准菜单。

8.3.2　菜单定制和标准图标选用

有时，制作者出于简洁、醒目的意图，并不希望在界面菜单条中出现那么繁杂的菜单项。此时，制作者就需要“定制”菜单。

例【8.3-2】　制作如图 8.3-2 所示的图形用户界面。该用户界面上，不仅配置了一个定制菜单，用以控制界面上坐标框是否封闭，而且还配置了一个“图形数据标识图标”。本例演示：定制菜单的制作步骤；菜单编辑器(Menu Editor)的使用及其回调子函数的编写；工具条编辑器(Toolbar Editor)的使用。

(1) 利用已有用户界面为基础制作新界面

- 在 MATLAB 工作平台上，点击工具图标，引出“GUIDE Quick start”对话窗。
- 在此对话窗“Open Existing GUI”页面上，选中 exm080301.fig 文件，再点击[Open]，引出显示 exm080301.fig 界面的界面编辑器。
- 在此界面编辑器上，选中{File > Save As}菜单项，另存为 exm080302.fig。

(2) 撤销原界面上的标准菜单和工具条

- 双击界面编辑器“设计工作区”的空白处，引出显示“图形窗”的属性编辑器。
- 把“MenuBar”的属性值设回“none”，把“ToolBar”设置为“auto”，就使标准菜单条及工具条从用户界面上消失。

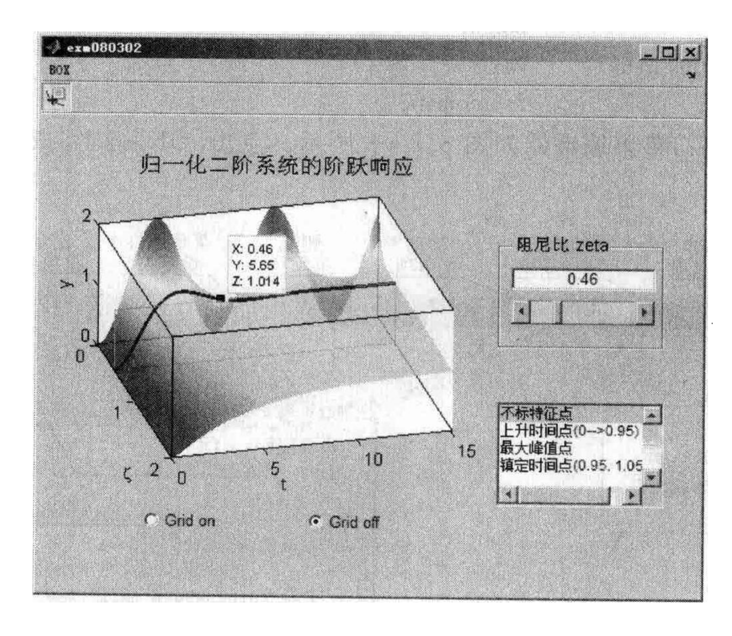


图 8.3-2　带定制菜单和选用标准工具图标的用户界面

(3) 定制菜单

● “(菜单)条菜单”的创建

■ 点击界面编辑器的工具图标，引出“菜单编辑器(Menu Editor)”，如图 8.3-3 所示。

■ 点击菜单编辑器上的图标后，在编辑器“Menu Bar”页左侧的菜单结构框里就出现未命名菜单“Untitled 1”。

■ 在菜单结构框里，选中未命名菜单，在编辑器右侧的“菜单属性(Menu Properties)”栏里，就会出现“需要填写的若干属性条目”。

■ 填写或勾选属性条目如下：

Label　　BOX　　%菜单名
Tag　　BOX1　　%菜单的句柄域名，即生成 handles.BOX1
Enable this item　　勾选　　%使该菜单可操作
Callback　　%automatic　　%这是默认设置；除非必须，一般不要更改

● “菜单项”的创建

■ 点击菜单编辑器的工具图标，就会在菜单结构框的“BOX”菜单下衍生出一个未命名菜单项“Untitled 2”。用光标在伴随产生菜单栏中，进行如下填写和勾选：

Label　　Box on
Tag　　boxon
Enable this item　　不勾选
Callback　　%automatic

■ 选中菜单结构框中的“BOX”菜单，再点击菜单编辑器的工具图标，就会在菜单结构框的“BOX”菜单下衍生出另一个未命名菜单项“Untitled 3”。在伴随产生菜单栏中，进行如下填写和勾选：

Label　　Box off

Tag　　　　　　　　　boxoff

Enable this item　　　勾选

Callback　　　　　　　%automatic

● 经以上操作后，菜单编辑器如图 8.3-3 所示。点击[OK]键，完成在菜单编辑器上的操作。

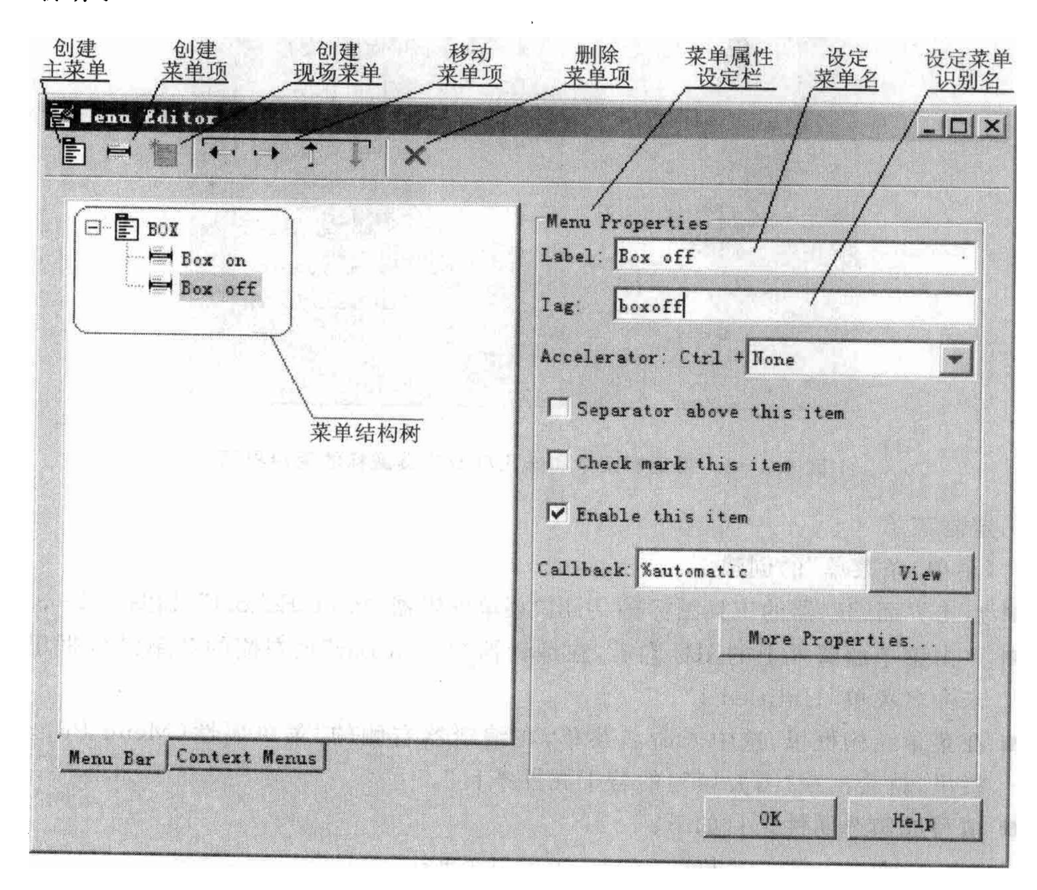


图 8.3-3　菜单编辑器的外形和功能分区

(4) 为定制菜单项编写回调子函数

在完成以上操作后，点击界面编辑器工具条上的▶图标，exm080302.fig 及 exm080302.m 文件分别得到更新。

此时，已经可以在生成的用户界面上看到菜单"BOX"。如果用鼠标点击该菜单，就会呈现包含两个选项的下拉菜单：一个是"失能"状态的"Box on"；另一个是"使能"的"Box off"。此时尽管外形已成，但如果用鼠标点击"Box off"，坐标框仍不会发生任何变化。原因是：这些菜单项的回调子函数尚未完成。

为使菜单项的相应功能实现，需要进行如下操作：

● 在 M 文件编辑器打开的 exm080302.m 文件中，可见在原先文件的最后增添了三个回调子函数：BOX1_Callback，boxon_Callback，boxoff_Callback。

● 在回调子函数 boxon_Callback 和 boxoff_Callback 中分别填写 M 码如下：

```
function boxon Callback(hObject, eventdata, handles)
```

```
%U Start ----------------------------------------U Start
boxon                                             %使坐标框封闭
set(handles.boxon,'Enable','off')                 %令"Box on"菜单项"失能"
set(handles.boxoff,'Enable','on')                 %令"Box off"菜单项"使能"
%U End ----------------------------------------U End

function boxoff Callback(hObject, eventdata, handles)
%U Start ----------------------------------------U Start
boxoff                                            %使坐标框不封闭
set(handles.boxoff,'Enable','off')                %令"Box off"菜单项"失能"
set(handles.boxon,'Enable','on')                  %令"Box on"菜单项"使能"
%U End ----------------------------------------U End
```

(5) 选用标准工具图标

MATLAB 图形窗的标准工具图标的配置步骤如下：

● 点击界面编辑器上的"工具条编辑"图标，引出如图 8.3-4 所示的工具条编辑器(Toolbar Editor)。

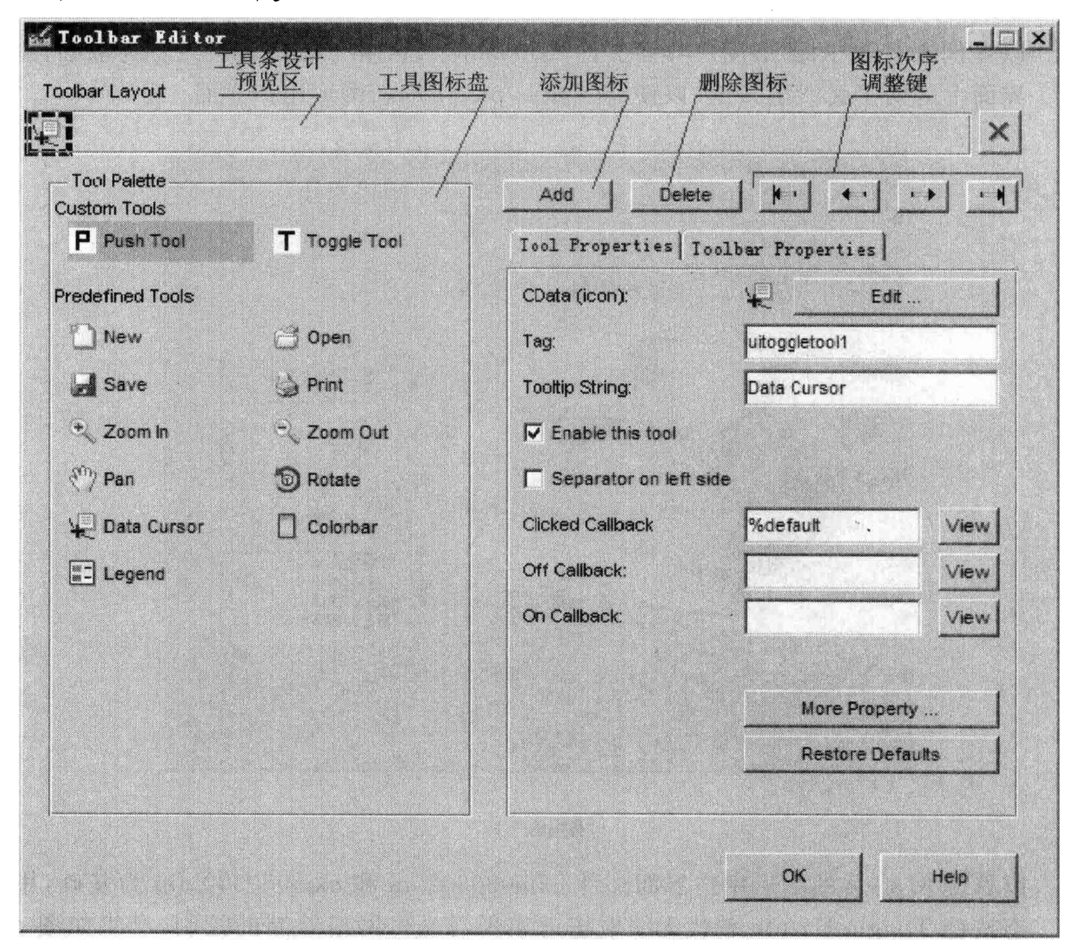


图 8.3-4　工具条编辑器的外形和功能分区

● 在图形窗标准工具图标盘(Tool Palette)中，点选图标，然后再单击"图标添加键"[Add]。于是，图标就出现在工具条设计预览区(Toolbar Layout)，如图 8.3-4。

(6) 保存及生成题目所要求的用户界面

在完成以上操作后，点击界面编辑器工具条中的▶图标，就能生成符合题目要求的界面，参见图 8.3-2。

说明

● handles 是界面编辑器创建用户界面时自动产生的一个"构架"变量。

● 在界面制作者给 GUIDE 产生的 M 执行文件添写指令之前，该构架变量的所有"域"用以存放界面组件(如窗、轴位框、控件、及菜单等)的句柄。

● 可以借助给 handles 增添的新域，存放"具体的应用数据"。

● 在各 GUIDE 生成的各子函数中，handles 各域存放的内容可直接读取，但若要保存更新了的 handles，就必须借助指令 guidata(hObject, handles)实现。

习题 8

1. 以算例 8.3-2 的图形用户界面文件 exm080302.m 和 exm080302.fig 为基础，删除原界面上的两个无线电按键，以便获得如图 p8-1 的新图形用户界面。

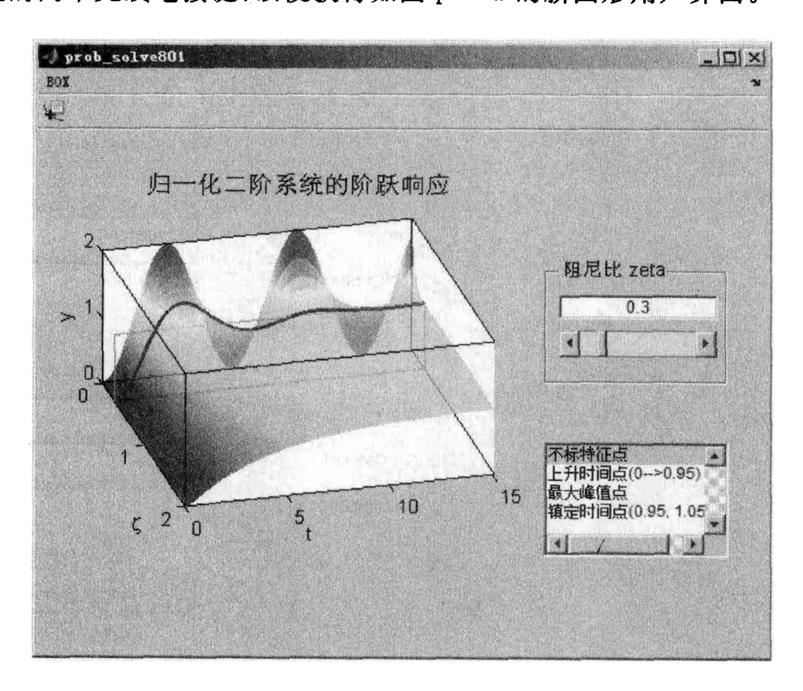


图 p8-1

2. 以算例 8.3-2 的图形用户界面文件 exm080302.m 和 exm080302.fig 为基础，用双稳态按键 Toggle Button 替代无线电按键实现对坐标网格绘制的控制，产生如图 p8-2 的新图形用户界面。

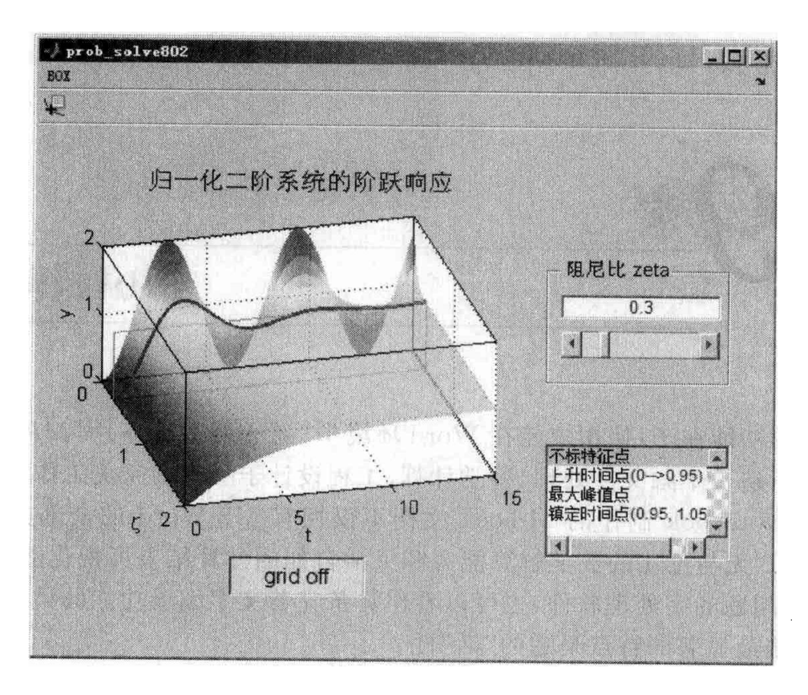

prob_solve802
BOX
归一化二阶系统的阶跃响应
阻尼比 zeta
0.3
不标特征点
上升时间点(0-->0.95)
最大峰值点
镇定时间点(0.95, 1.05
grid off

图 p8-2

第9章 Notebook

Notebook 的功能在于:使用户能在 Word 环境中“随心所欲地享用”MATLAB 浩瀚的科技资源,为用户营造一个融文字处理、科学计算、工程设计于一体的完美工作环境。

MATLAB Notebook 制作的 M-book 文档不仅拥有 MS-Word 的全部文字处理功能,而且具备 MATLAB 无与伦比的数学解算能力和灵活自如的计算结果可视化能力。它既可以看作解决各种计算问题的字处理软件,也可以看作具备完善文字编辑功能的科技应用软件。

M-book 文档最显著的特点是它的“活”性。

- 它为论文、科技报告、讲义教材和学生作业的撰写营造了文字语言思维和科学计算思维的和谐环境。
- 用 M-book 写成的电子著作、电子文稿和讲义教材不仅图文并茂,而且动静结合。那些由 MATLAB 指令构成的例题、演示,都可供读者亲自操作,举一反三,从而在“手脑并用”的环境中由此及彼、由浅入深。

9.1 Notebook 的配置和启动

9.1.1 Notebook 的配置

(1) 与 MATLAB 适配的 Word

随 MATLAB 版本的升级,与其适配的版本也会发生变化。以 MATLAB R2010a 为例,能用来配置 Notebook 环境的 Word 版本是 2002,2003,2007。

(2) Notebook 的配置

在 Windows 已经装有前述 Word 版本的前提下,在 MATLAB 中配置 Notebook 环境十分简便。只要在 MATLAB 指令窗中运行以下指令,配置过程将自动进行。

```
notebook -setup
```

假如指令窗中出现如下信息,就表示配置成功。

```
Setup complete
```

9.1.2 Notebook 的启动

1. 创建新的 M-book 文件

(1) 在 Word 默认窗口(即 Normal. dot)下创建新的 M-book 文档

- 选择 Word 窗口的下拉菜单项{文件:新建}。
- 在弹出的对话框中,选择"M-book"模板,按[确定]键。
- 于是,Word 的窗口由原先的默认式样变成 M-book 式样,如图 9.1-1 所示,自动开启一个新的 MATLAB 作为其服务器,而不管此前 Windows 平台上是否已经开启了 MATLAB。
- 注意:由点击 M-book 模板引出的 MATLAB 当前目录是"MATLAB 程序所在的目录",比如 MATLAB R2010a;而由点击 MATLAB 图标引出的 MATLAB 当前目录是用户专用目录 C:\Documents and Settings\user\My Documents\MATLAB。

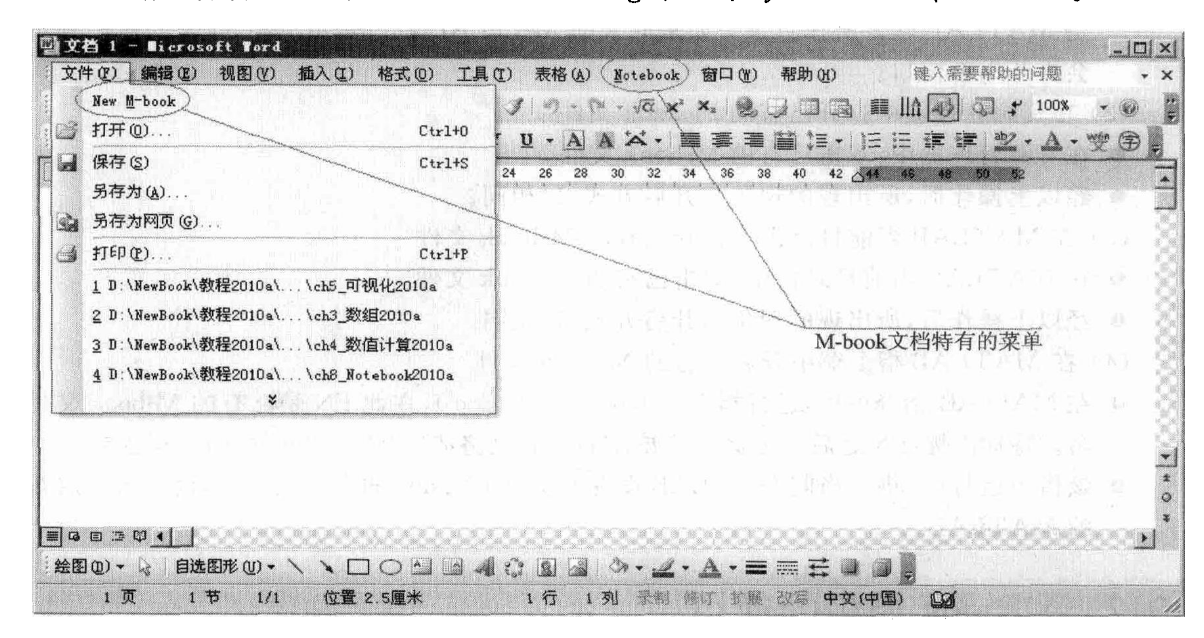


图 9.1-1　新建 M-book 文档的界面

(2) 在 MATLAB 中创建新的 M-book 文件

在 MATLAB 指令窗中运行以下指令,都可以创建新的 M-book 文件,即

notebook	引出一个未命名的 M-book 文档界面
notebook NewFileName	在当前目录上创建名为 NewFileName 的空白文件(此指令应慎用!)

说明

- 第一个指令引出的是使用 M-book.dot 模板的未命名文档界面,参见图 9.1-1。
- 第二个指令不仅引出 M-book 界面,而且自动在当前目录上产生一个名为 NewFileName.doc 的空白文件。
- 第二个指令的使用要特别小心。要特别注意:NewFileName 千万不要与当前目录上已经存在的文件同名,否则,将把扩展名为 doc 的原有同名文件改变为一个空白文件,而且发生这种改变前没有任何提示可控操作。
- 以上指令运行后,不会引出新的 MATLAB,而只是把原有的 MATLAB 用作自动服务器,此时在 MATLAB 指令窗中会出现"Warning: MATLAB is now an automation server"的信息。

2. 打开已有的 M-book 文件

(1) 在 Word 默认窗口下打开已有的 M-book 文件

- 在 Word 默认的窗口下打开已有 M-book 文件的方法与打开一般 Word 文件没有两样。最常用的方法是选中下拉菜单项 {文件:打开},然后从弹出的对话框中选择所需要编辑的 M-book 文件。
- 以上指令运行后,自动开启一个新的 MATLAB 作为 M-book 的服务器,而不管此前 Windows 平台上是否已经开启了 MATLAB。新开启的 MATLAB 窗口的当前目录是 MATLAB 软件所在目录。如果原来的 MATLAB 窗口是由 notebook 开启的则不会再开出新窗口。

(2) 在资源管理器中打开已有的 M-book 文件

- 在资源管理器中,双击已有的 M-book 文件。
- 经以上操作后,所出现的现象与开启方式(1)相同。

(3) 在 MATLAB 当前目录窗中打开已有的 M-book 文件

- 在 MATLAB 当前目录窗中,双击已有的 M-book 文件。
- 经以上操作后,所出现的现象与开启方式(1)相同。

(4) 在 MATLAB 指令窗中开启已有的 M-book 文件

- 在 MATLAB 指令窗中,运行指令 notebook('FN. doc'),在此 FN 是已有的 M-book 文件名。特别提醒:FN 之后一定要带扩展名. doc,否则将造成"清空原文件"的严重后果。
- 该指令运行后,将把当前 MATLAB 设置成文件 FN. doc 的自动服务器,而不再开启新的 MATLAB。

9.2 M-book 模板的使用

M-book. dot 模板的外形和使用方法,几乎与普通 Word 模板 Normal. dot 完全相同。因此,在 M-book 中,文字、图像、表格、数学公式等的输入、排版、编辑方法,与在普通 Word 文档中没有什么区别。

M-book 的特点在于:该模板以 MATLAB 为其计算服务器。这些特殊功能集中地反映在 {Notebook} 下拉菜单中,如图 9.2-1 所示。

9.2.1 输入细胞(群)的创建和运行

1. 细胞(群)

在 Notebook 中,凡参与 Word 和 MATLAB 之间信息交换的部分,就称之为"细胞(群)(cells or cell group)"。由 M-book 送向 MATLAB 的指令,称为"输入细胞(Input cells)";由 MATLAB 返回 M-book 的计算结果,称为"输出细胞(output cells)"。

(输入)细胞和(输入)细胞群没有根本的区别,也没有必要刻意区别。输入、输出细胞不必成对生存:输入细胞可以单独存在,但输出细胞必须依赖输入细胞而存在。

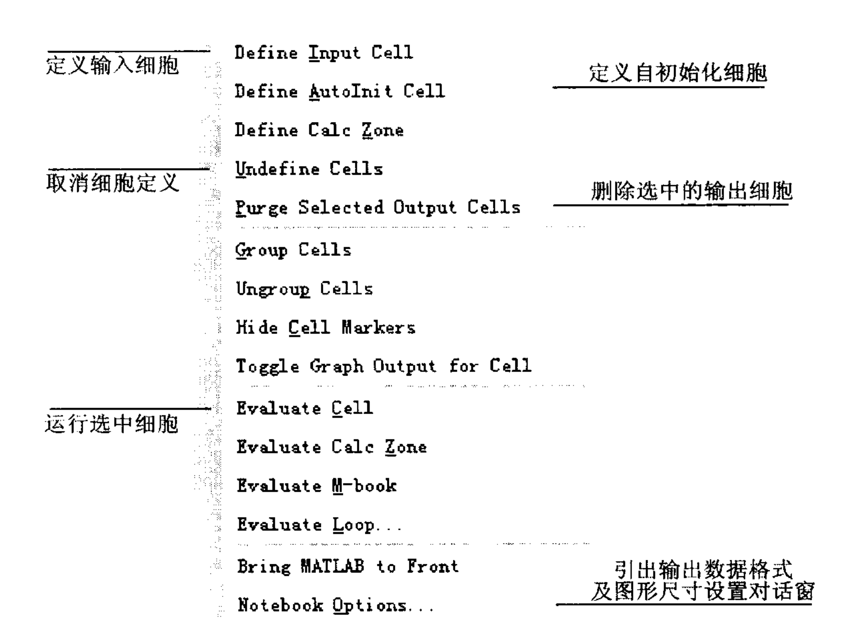


图 9.2-1　{Notebook}下拉菜单的常用功能项

2. 基本操作

较之普通 Word，Notebook 最宝贵的东西就是输入细胞(群)。从应用上讲，学会了输入细胞(群)的创建和运行，就意味着掌握了 Notebook。

细胞(群)创建、正确运行的两个基本操作：

- 以普通文本形式输入的必须是 MATLAB 指令。特别注意：标点符号必须是在英文状态下输入的。
- 不管文本形式的一条指令有多长，不管一行有多少条文本形式指令，不管有多少行文本形式指令，只要能用鼠标把它们同时“点亮”选中，都可以被创建或运行，具体如下。

  ① 在文本内容“点亮”后，按组合键 [Ctrl-Enter]，或选中下拉菜单项 {Evaluate Cell}，那么被“点亮”部分就被激活成输入细胞(群)，文字颜色将呈现为绿色。与此同时，细胞所含指令被送进 MATLAB 运行，最后在该输入细胞的下方嵌入计算结果(数据或图形)。这就是输出细胞，它的文字用蓝色显示。

  ② 在文本内容“点亮”后，按组合键 [Alt-D]，或选中下拉菜单项 {Define Input Cell}，被“点亮”部分只是变成了输入细胞(群)，并没有运行，当然也就没有运行结果。

3. 输入细胞(群)操作示例

例【9.2-1】 演示：创建并运行输入细胞的基本操作方法。

(1) 输入细胞的单纯生成法

在“正文”段落里，英文状态下，按普通的文本输入方式，键入如下一行指令；用鼠标把该行内容“点亮”；然后按组合键 [Alt-D]，文本形式的指令就变成了绿色的输入细胞，但并不送去

运算,当然也不会输出任何结果。

```
xx=(1:5)/5*pi;yy=sin(xx).*exp(xx)
```

(2) 输入细胞生成、运行同时进行的操作方法

在英文状态下,在"正文"段落里,按普通的文本输入方式,键入如下一行指令;用鼠标把该行内容"点亮";然后按组合键 [Ctrl-Enter];于是该指令就会自动变成绿色输入细胞,并给出蓝色运算结果,即输出细胞。

```
x=(1:4)/4*pi;y=sin(x).*exp(x)
y =
    1.5509    4.8105    7.4605    0.0000
```

说明

- 在 M-book 中,最常用且最可靠的操作是:"点亮"待运算指令,按组合键 [Ctrl-Enter]。它的功能是:产生并运行当前细胞(群)。
- 在中文 M-book 文档中,特别注意:不要把中文标点混杂在 MATLAB 指令中。否则,易产生运行错误,甚至造成死机。

例【9.2-2】 演示:生成完整图形的所有指令必须定义在同一细胞(群)中(见图 9.2-2)。

```
t=0:0.1:10;y=1-cos(t).*exp(-t);                                   % <1>
tt=[0,10,10,0];
yy=[0.95,0.95,1.05,1.05];
fill(tt,yy,'g'),axis([0,10,0,1.2]),xlabel('t'),ylabel('y')       % <4>
hold on                                                            % <5>
plot(t,y,'k','LineWidth',4)                                        % <6>
hold off                                                           % <7>
ymax=max(y)                                                        % <8>
ymax =
    1.0669
```

说明

- 使上述程序正确运行的最简单的方法是:用鼠标把从〈1〉到〈8〉的指令全部"点亮",然后按组合键 [Ctrl-Enter],就能得到正确结果。
- 保证程序正确运行的起码条件是:指令〈4〉〈5〉〈6〉应该在同一个输入细胞(群)中。否则会产生多幅图形,其中只有最后一幅是完整的。
- 不管输入细胞中显示运算结果的指令次序如何,在输出细胞中,非图形结果(包括数值、字符和符号对象等)总安排在图形的前面显示。

9.2.2 Notebook 菜单的其他选项

1. 自初始化细胞及其应用

(1) 自初始化细胞

自初始化细胞(autoInit cell)与输入细胞功能的唯一不同是:当用户启动一个 M-book 文件时,包含在该文件中的自初始化细胞会自动被送去运算。而输入细胞不具备这种功能。若

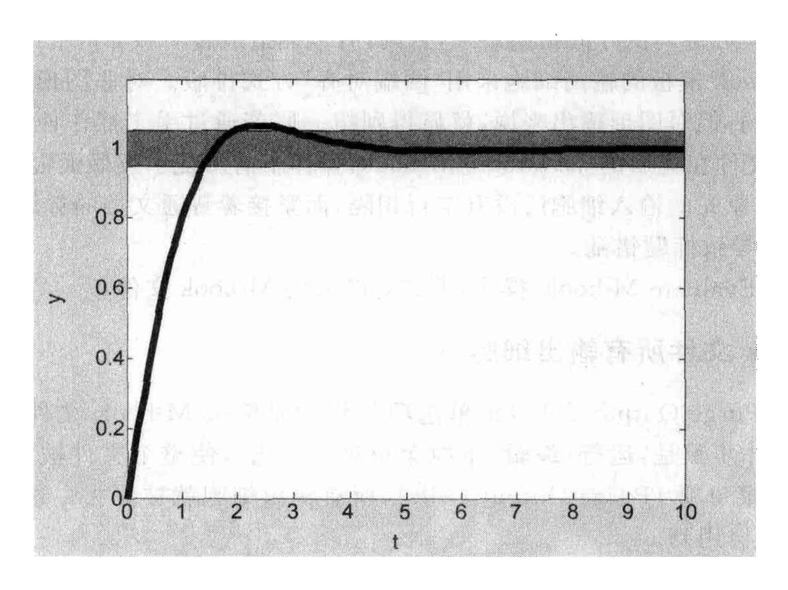


图 9.2-2 细胞群产生的完整图形

用户需要在打开文件时，对 MATLAB 工作内存进行初始化工作，那么自初始化细胞就特别有用。

自初始化细胞有两种来源：一是，文本形式的 MATLAB 命令；二是，已经存在的输入细胞。为把它们变成自初始化细胞，先用鼠标选亮它们，然后选用 {Notebook:Define AutoInit Cell}菜单选项即可。

(2) 工作内存的初始化

M-book 所有计算都在 MATLAB 中进行，参与运算的所有变量都储存在 MATLAB 工作内存中。各 M-book 文件和 MATLAB 指令窗分享同一个"计算引擎(compute engine)"和同一个工作内存。工作内存中的变量是各 M-book 文件和 MATLAB 指令窗工作后共同产生的。对此，用户应有清醒的认识。记住这条工作原理，将能使 M-book 文件的运用灵活自如。

当用户同时打开几个 M-book 文件，或在 MATLAB 指令窗和 M-book 文件间交互运作时，要特别注意不同文件和窗口之间变量的相互影响。假如要保证某 M-book 文件独占 MATLAB 工作内存，保证该文件的输入/输出数据间的一致性，一个有效的办法是：把 clear 定义为该文件的第一个自初始化细胞。

2. 整个 M-book 文件的运行

{Notebook}菜单中的 {Evaluate M-book}选项可以运行整个 M-book 文件，即把文档中所有输入细胞送到 MATLAB 中去运行。不管光标处在该文档的什么地方，运行总是从文件首部开始。在整个 M-book 文件运行时，它不但会把所有原输出细胞中的内容刷新；而且会补写新的输出细胞。这个命令在保证整个 M-book 文件中所有指令、数据和图形的一致性方面十分有用。

在此提醒用户注意：假如原 M-book 文档的输出细胞自生成后没有再经历过编辑(如图形"对中"、输出细胞位置的前后"搬动"和输出细胞前后空行的"删除"等)操作，那么使用 {Eval-

uate M-book}选项可得到良好的预期效果，否则，有可能造成整个版面的混乱。

实际上，M-book模板的输出细胞采用“两端对齐”方式排版。对非图形输出来说，这是符合普通排版习惯的；但对图形输出来说，就显得别扭。假若通过手工操作使图形“对中”，这样形成的M-book文件在此后的{Evaluate M-book}操作下有可能引发版面混乱。

此外，假如在原先的输入细胞后没有空行相隔，而紧接着普通文档的标题，那么{Evaluate M-book}操作，将导致标题错乱。

总之，慎用{Evaluate M-book}操作，尤其对较大的M-book文件。

3. 删去M-book文件所有输出细胞

{Notebook:Purge Output Cells}菜单选项的作用是删去M-book文件中的所有输出细胞。它的具体操作步骤是：运行{编辑}下拉菜单项{全选}，使整个文件被选中；然后再运行{Notebook}下拉菜单项{Purge Output Cells}，所有输出细胞就被删去。这个指令在撰写报告、布置作业时常会用到。

9.2.3 输出细胞的格式控制

输出细胞容纳MATLAB的各种输出结果：数据、图形、错误信息。输出数据的有效数字、图形的大小都可以借助如图9.2-3所示的对话框加以控制。打开控制对话框的方法是：选中{Notebook}菜单中{Notebook Options}选项。

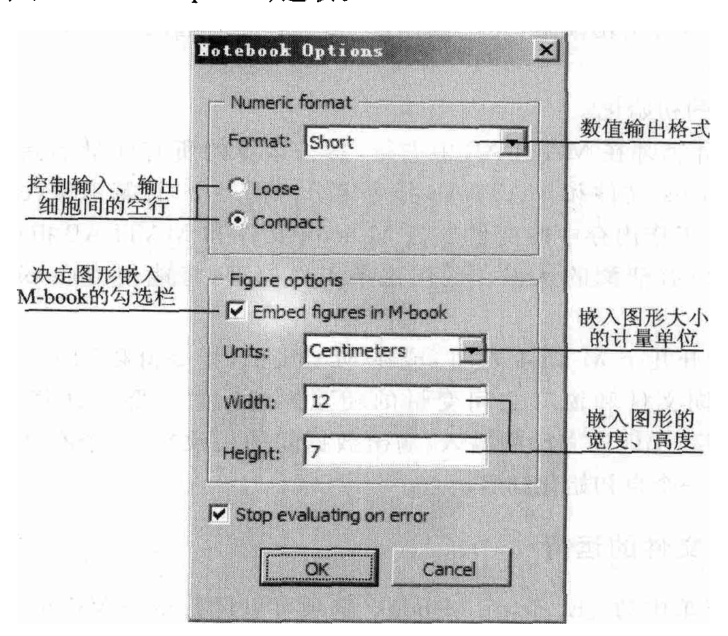


图9.2-3 控制输出细胞格式的对话框

1. 输出数据的表示法

在MATLAB指令窗中，format指令所能控制的输出数据格式，在Notebook中同样存在，如：Short，Long，Hex，Bank，Short e，Long e等。

在 M-book 中，输出细胞中数据的表示方式有 3 种控制方法。

（1）利用图 9.2－3 所示对话框中{Numeric Format}子框下拉列表进行设置。

（2）借助 M-book 输入细胞中的 format 指令进行设置。

（3）在 MATLAB 指令窗中，用 format 指令进行设置。

以上 3 种方法的效果相同，读者应视情况选用。

2. 输出数据间的空行控制

图 9.2－3 所示对话框中的{Loose}和{Compact}选项用来控制输入细胞与输出细胞之间的空白区间。比如，选择{Loose}后，在 M-book 文档的输入细胞和输出细胞之间加入一个空行。注意：这种控制方法与输入细胞群中的 format loose，format compact 命令有不同的功能。后者控制的将是输出细胞与输出细胞之间空行。

在此须再强调，不同输出格式给出不同的数据显示精度，但内部存储及运算都是以相同的双精度进行的。

3. 图形的嵌入控制

在默认情况下，图 9.2－3 所示对话框中的"镶嵌选项"{Embed Figures in M-book}处于"勾选"状态。此时，输出图形将被镶嵌在 M-book 文档中。假如不勾选"镶嵌选项"，那么在 M-book 文档中，将肯定没有输出图形。"镶嵌选项"的控制作用，将影响其后运行的所有输入细胞中的绘图输出。

例【9.2－3】 在同一细胞群中虽然包含绘制两幅图形的指令，但实际上只能把最后一个绘制的图形镶嵌进 M-book（见图 9.2－4）。

```
surf(peaks)                          % 绘制曲面图
colormap(hot)
t = (0:50)/50 * pi;y = sin(t);
plot(t,y)                            % 绘制曲线图
```

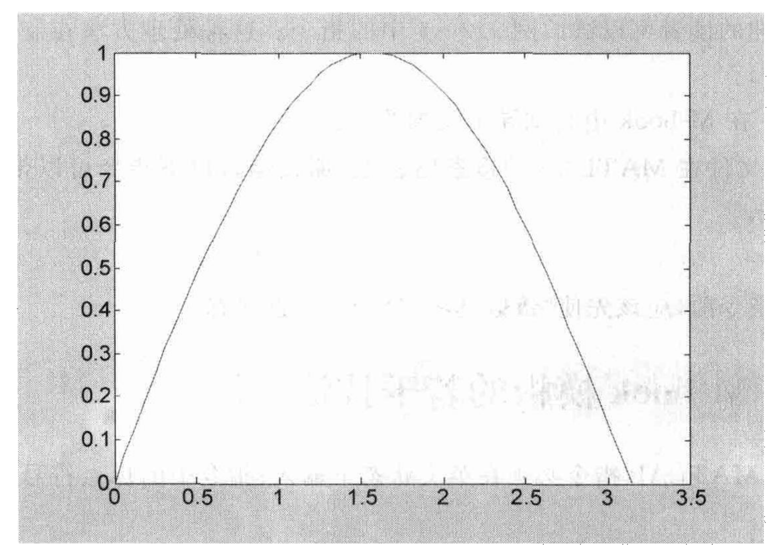


图 9.2－4　同一细胞群中最后一幅绘制的曲线图

4. 嵌入图形大小的控制

在图 9.2-3 所示的 Notebook Options 对话框下方有三个栏目:Units,Width,Height。用户可通过对这三栏的设置,决定嵌入图形框的大小。

注意:由于嵌入图形框的大小由人为指定,一般不会顾及其宽高比是否与 MATLAB 图形窗相同。这样,诸如 axis square ,axis equal 等控制宽高比的指令将不能发挥原先的作用。

图形一旦被嵌入 M-book,就像普通 Word 图形一样,可以被移动、缩放、剪裁和编辑。有关内容,请读者查阅 Word 书籍。

5. 嵌入图形的背景色问题

在默认情况下,正常嵌入图形的背景色应是灰/白的。假如由于某种原因,所嵌图形出现灰/黑背景色,那么可采取以下两种措施的一种,尝试纠正。

- 打开 Notebook Options 对话框,确认"镶嵌选项"处于勾选状态,并再次单击[OK]键。然后,再重新运行输入细胞。
- 在 MATLAB 指令窗中,运行`whitebg('white')`,或运行`close;colordef white`,然后再重新运行输入细胞。

6. M-book 处理活动画面的能力

M-book 输出细胞中图形的产生机理是:在模板"宏"的控制下,输入细胞中的绘图指令被送去 MATLAB 计算后,MATLAB 依然是在图形窗中产生图形,只有该窗中的最终画面才被复制到剪贴板,并再从那里被粘贴到 M-book。

由此机理可知,M-book 模板既不可能表现活动画面,也不可能用于画面的交互操作。正因为如此,本书每凡涉及动画和交互画面时,都一再提醒读者,那些指令应在 MATLAB 指令窗中运行。

尽管如此,仍可以从 M-book 发出动画指令,把图形窗带到前台,在图形窗中表现画面的活动情景。有兴趣的读者可以试试例 9.2-4 中的指令。这种处理方法在制作计算机演讲稿时,有可能会用到。

例【9.2-4】 在 M-book 中尝试导出动画图形。

假若下列 M 文件在 MATLAB 的搜索路径上,那么运行以下指令可以带出动画图形窗,即`anim zzy1(1),shg`。

说明

在运行以上指令前,应该先使"镶嵌选项"处于不勾选状态。

9.3 使用 M-book 模板的若干技法

- 文档中的 MATLAB 指令必须在英文状态下输入;指令中的标点符号必须在英文状态下输入。

- 续行号不能用于输入细胞。
- 不管一条指令多长，只要不用“硬回车”换行，总可以被鼠标全部“点亮”并按组合键[Ctrl-Enter]后正确地运行。
- MATLAB 指令在 M-book 中的运行速度比在指令窗中慢得多。因此，符号计算指令、编译指令等在 M-book 中运行时，有可能发生“运行时间过长”或“出错”的警告。遇到这种情况，用户最好还是让那些指令在指令窗中直接运行。

附录A 字符串、胞元及构架数组

A.1 字符串数组

字符串数组在 MATLAB 中不很重要，但也不可缺少。假如没有串数组及相应的操作，那么数据可视化将会遇到困难，构造 MATLAB 的宏指令也将会遇到困难。

字符变量的创建方式是：在指令窗中，先把待建的字符放在“单引号对”中，按[Enter]键。注意，这种“单引号对”必须在英文状态下输入。“单引号对”是 MATLAB 识别送来内容“身份”(是变量名、数字，还是字符串)所必需的。

例【A.1-1】 数值量与字符串的区别。

```
clear                        % 清除所有内存变量
a = 12345.6789               % 给变量 a 赋数值标量
class(a)                     % 对变量 a 的类别进行判断
a_s = size(a)                % 数值数组 a 的“大小”
a =
  1.2346e + 004
ans =
double
a_s =
     1      1
b = 'S'                      % 给变量 b 赋字符标量(即单个字符)
class(b)                     % 对变量 b 的类别进行判断
b_s = size(b)                % 符号数组 b 的“大小”
b =
S
ans =
char
b_s =
     1      1
whos                         % 观察变量 a,b 在内存中所占字节
  Name      Size          Bytes  Class     Attributes
  a         1x1               8  double
  a_s       1x2              16  double
```

```
  ans         1x4              8   char
  b           1x1              2   char
  b_s         1x2             16   double
```

例【A.1－2】 串的基本属性、标识和简单操作。

（1）创建串数组

下面指令创建一个由 19 个字符组成的串。这 19 个字符必须被放在“单引号对”内。

```
a = 'This is an example.'
a =
This is an example.
```

（2）串数组 a 的大小

在以上赋值后，变量 a 就是一个串数组。该串的每个字符（英文字母、空格和标点都是平等的）占据一个元素位。该串数组的大小可用下面指令获得。

```
size(a)
ans =
     1    19
```

（3）中文字符串数组

中文字符创建时一定要特别注意：那字符外面的“单引号对”必须在英文状态下输入，即必须是英文的“单引号”而不能是中文“单引号”。

```
A = '这是算例。'                                   % 创建中文字符串
A =
这是算例。
```

（4）由小串构成长串

```
ab = [A(1:4),'A.1 - 2',A(5)]                % 字符串元素也能援引
ab =
这是算例 A.1 - 2。
```

例【A.1－3】 实现数值向字符串转换的函数int2str，num2str。

（1）int2str 把整数数组转换成串数组（非整数将被四舍五入圆整后再转换）

```
A = eye(2,4);                  % 生成一个(2 × 4)数值数组
A_str1 = int2str(A)            % 转换成(2 × 10)串数组。请读者自己用 size 检验
A_str1 =
1  0  0  0
0  1  0  0
```

（2）num2str 把非整数数组转换为串数组（常用于图形中，数据点的标识）

```
rand('twister',0)
B = rand(2,4);                    % 生成数值矩阵
B3 = num2str(B,3)                 % 保持 3 位有效数字，转换为串
B3 =
 0.549    0.603    0.424    0.438
 0.715    0.545    0.646    0.892
```

例【A.1-4】 综合例题:在 MATLAB 计算生成的图形上标出图名和最大值点坐标。(见图 A.1-1)

```
clear                          % 清除内存中的所有变量
a=2;                           % 设置衰减系数
w=3;                           % 设置振荡频率
t=0:0.01:10;                   % 取自变量采样数组
y=exp(-a*t).*sin(w*t);                    % 计算函数值,产生函数数组
[y_max,i_max]=max(y);                     % 找最大值元素位置
t_text=['t=',num2str(t(i_max))];          % 生成最大值点的横坐标字符串          <7>
y_text=['y=',num2str(y_max)];             % 生成最大值点的纵坐标字符串          <8>
max_text=char('maximum',t_text,y_text);   % 生成标志最大值点的三行字符串        <9>
tit=['y=exp(-',num2str(a),'t)*sin(',num2str(w),'t)'];
                                          % 生成标志图名用的字符串              <11>
plot(t,zeros(size(t)),'k')                          % 画纵坐标为 0 的基准线
hold on                                             % 保持绘制的线不被清除
plot(t,y,'b')                                       % 用兰色画 y(t)曲线
plot(t(i_max),y_max,'r.','MarkerSize',20)           % 用大红点标最大值点
text(t(i_max)+0.3,y_max+0.05,max_text)              % 在图上书写最大值点的数据值   <16>
title(tit),xlabel('t'),ylabel('y')                  % 书写图名、横坐标名、纵坐标名
hold off
```

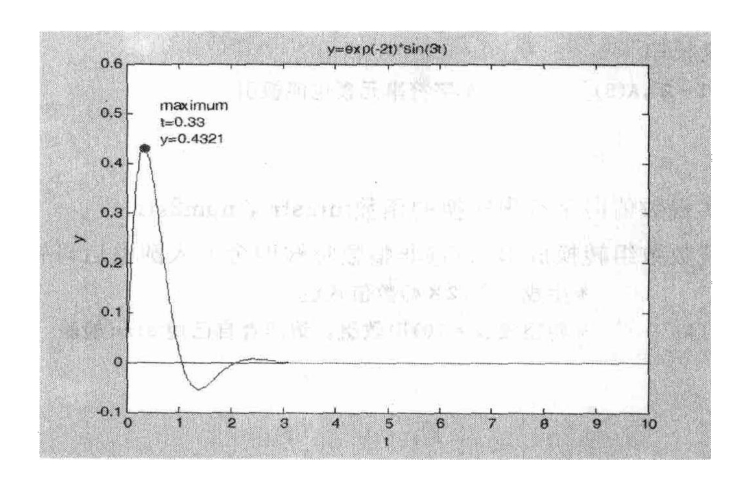


图 A.1-1 字符串运用示意图

说明

- 本例第<7><8>句是 num2str 指令的一种典型运用。由这种方式组成的字符串的特点是:由数值转换而得的那部分字符是可以随计算所产生的数据而变。第<11>句也属这种类型,它使得图名中的衰减系数 a 和振荡频率 w 可随不同的赋值而变(本例 $a=2$, $w=3$)。
- 本例第<9>句把多个字符串变成一个"多行字符串",供第<16>句调用。

A.2 胞元数组

许多大银行都有一个管理十分完善的保险箱库。这保险箱库的最小单位是箱柜，可以存放任何东西（如珠宝、债券、现金、文件等）。每个箱柜被编号，一个个编号的箱柜组合成排，一排排编号的箱柜排组合成室，一个个编号的室便组合成那银行的保险箱库。

胞元数组（Cell array）如同银行里的保险箱库一样。该数组的基本组分（element）是胞元（Cell）。每个胞元本身在数组中是平等的，它们只能以下标区分。同一个胞元数组中不同胞元可以存放不同类型和不同大小的数据，如任意维数值数组、字符串数组、符号对象等。

注意区分胞元和胞元内容是两个不同概念。正由于此，有两种不同操作：

- “胞元标识（Cell Indexing）”，例如 $A(2,3)$ 是指 A 胞元数组中的第 2 行第 3 列的胞元；
- “胞元内容编址（Content Addressing）”，如 $A\{2,3\}$ 是指 A 胞元数组第 2 行第 3 列胞元中所存放的内容。请注意：花括号{ }的用法。

例【A.2-1】 本例演示：（2×2）胞元数组的创建；同一个胞元数组中的不同胞元可以存放不同类型、不同大小的数据。

```
clear
C str = '这是胞元数组创建算例 1';                %产生字符串
R = reshape(1:9,3,3);                          %产生(3×3)实数阵 R
Cn = [1 + 2i];                                 %产生复数标量
S sym = sym('sin( - 3 * t) * exp( - t)');      %产生符号函数量

%创建胞元数组方法之一
B{1,1} = C str;                                                      %<6>
B{1,2} = R;
B{2,1} = Cn;
B{2,2} = S sym;                                                      %<9>

%胞元的援引
a = B(1,2)                        %注意，这里用"圆括号"
class(a)
a =
    [3x3 double]
ans =
cell

%胞元内容的援引
b = B{1,2}                        %注意，这里用"花括号"
class(b)
b =
     1     4     7
     2     5     8
     3     6     9
ans =
double
```

说明

- 第<6>到第<9>条指令是创建胞元数组的方法之一，此法常用于小型胞元数组。胞元数组更有效的创建方法是，借助 cell 指令预定义胞元空数组。
- 注意 *a* 是胞元，*b* 是(3×3)的双精度矩阵。

A.3 构架数组

与胞元数组一样，构架数组(structure array)也能在一个数组里存放各类数据。从一定意义上讲，构架数组组织数据的能力比胞元数组更强、更富于变化。

构架数组的基本组分(element)是构架(structure)。数组中的每个构架是平等的，它们以下标区分。构架必须在划分"域"后才能使用。数据不能直接存放于构架，而只能存放在域中。构架的域可以存放任何类型、任何大小的数组(如任意维数值数组、字符串数组、符号对象等)。而且，不同构架的同名域中存放的内容可以不同。

特别注意：构架名和域名之间的小黑点"."的作用。

例【A.3-1】 通过温室数据(温室名、容积、温度和湿度等)演示：单构架的创建和显示。

(1) 直接对域赋值法产生"单构架"，即(1×1)构架数组。

```
clear
G.name = '一号房';                       % G构架的 name 域存放字符串。            <1>
G.volume = 2000;                         % G构架的 volume 域存放标量数值          <2>
G.temperature = [31.2,30.4,31.6,28.7];
                                         % G构架的 temperature 域存放一维数值数组 <3>
G.humidity = [62.1,59.5,57.7,61.5;63,60,58.1,62.3];
                                         % G构架的 humidity 域存放二维数值数组    <4>
```

(2) 向第二个构架的域赋值，形成构架数组。

```
G(2).name = '二号房';
G(2).volume = 2400;
```

(3) 构架数组的显示

```
G                                        % 显示构架数组                           <7>
G =
1x2 struct array with fields:
    name
    volume
    temperature
    humidity
```

(4) 构架元素的显示

```
G(1)
ans =
          name: '一号房'
        volume: 2000
   temperature: [31.2000 30.4000 31.6000 28.7000]
```

```
      humidity: [2x4 double]
```

(5) 构架域的显示

```
G.humidity                                                          %      <9>
ans =
   62.1000    59.5000    57.7000    61.5000
   63.0000    60.0000    58.1000    62.3000
ans =
[]
```

说明

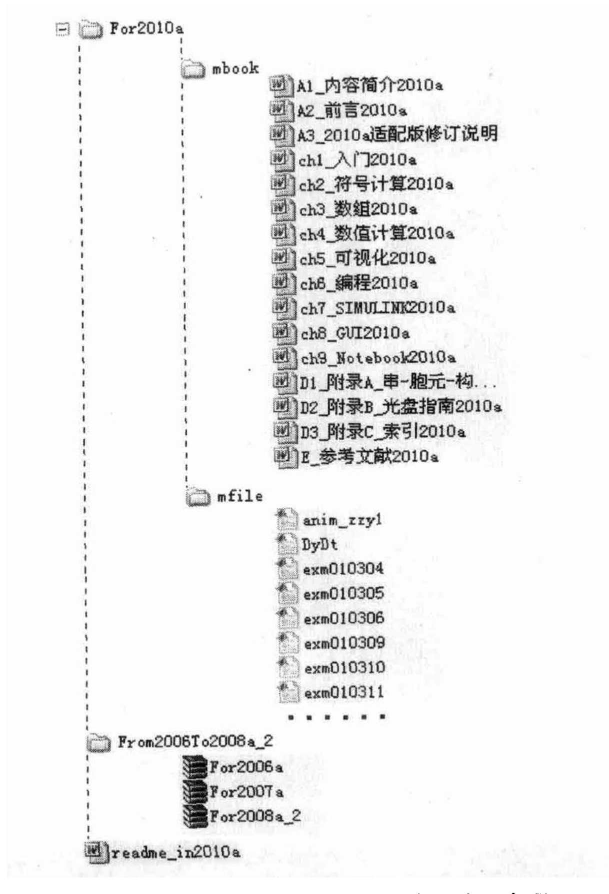
- 对于非单构架，直接键入单构架名（比如指令<7>），通常只能得到该构架的结构信息，而不显示该构架域中的具体内容，除非该构架域中的内容是极为简单的数值变量或单行字符串。
- 指令窗中键入构架域名时（如指令<9>），可显示各构架的该域内容。

附录B 光盘使用说明

B.1 光盘文件的结构

在光盘上有两个文件夹。

- 光盘上的目录结构和各种文件的存放位置如图 B.2－1 所示。
- {For2010a}目录包含如下两个子目录：
  - ■ n mbook　　存放着包含本书全部算例的 M－book 形式 DOC 文档。
  - ■ n mfiles　　汇集了全书 90％以上算例的 M 文件、MDL 文件和 FIG 文件。



图B.2－1　光盘上的目录结构与文件存放示意图

B.2　光盘对软件环境的要求

- 需要 Offiece 2003，XP 和 MATLAB R2010a 支持。（书中除 Simulink 模型 MDL 文件外，其他绝大多数指令和文件在 MATLAB 其他新老版本中也都能运行）。
- 假如要完好运行 mbook 文件夹上的 DOC 文件，则需要 Notebook 环境。（关于 Notebook 环境的设置参见第 8 章）。

B.3　光盘文件的操作准备

在运行光盘文件之前，应首先把\mfiles 文件夹设置为当前目录或设置在 MATLAB 的搜索路径上（具体方法参见第 1.6 节）。

B.4　M-book 文件夹上 DOC 文件的使用

光盘中的 DOC 文件在 MATLAB R2010a 的 Notebook 环境中生成，DOC 文件各章节的编号、名称与印刷版一致。

(1) 光盘 DOC 文件的功用

- 弥补了印刷版丢失的色彩信息

  在 MATLAB 中，用 M 文件编辑器或 Notebook 编写的指令或文件运行的结果（尤其图形）都采用不同的色彩鲜明地表现对象特征。但目前印刷版书籍出于价格和技术原因，不得不牺牲色彩信息而采用“黑白”处理。读者借助本光盘可克服印刷版丢失色彩信息的缺憾。

- 为教师制作本教材电子讲稿提供模板

  据作者 10 多年 MATLAB 的教学经验发现，在 MATLAB 的课堂教学中，电子讲稿最好使用 M-book 模板制作，而不宜采用 PowerPoint 制作。出于这样的考虑，本书光盘提供了各章的 DOC 文件。它们包含完整的章节结构和名称，包含所有算例的题解要求和完整的解题程序。

  主讲这门课程的老师，可以根据自己的经验和心得，针对具体教学对象，通过对本光盘 DOC 文件进行适当的剪裁，增补少量提示、警告和归纳性文字，就可得到适宜的电子讲稿。

- 为学生完成电子作业提供模板

  学生平时做作业最好使用 M-book 模板完成。一，光盘电子版准确描述了习题的要求（如图形色彩、光照等）；二，在 M-book 文件中，能把文字解释、MATLAB 指令、计算结果及图形有机地结合在一起。

- 为习题提供答案

  由于有些习题答案（如图形等）必须通过光盘电子版才能较准确地表达，所以本书相关章节的习题答案放在光盘中。

- 为读者提供了与印刷版对应的 Notebook 演练环境

本光盘中 DOC 文件的章节结构、算例编号与印刷版完全相同。因此在学习过程中，读者可在本光盘启动的 Notebook 环境中，直接运行算例，观察运行结果；或改变若干指令，举一反三地观察运行结果的变化；或通过简单的复制操作，使相应指令在 MATLAB 指令窗中运行，从而避免自己键入的错误。

（2）DOC 文件的开启

所有 DOC 文档都是在 Word＋MATLAB R2010a 构成的 Notebook 环境中生成的。因此，在相同环境下开启是最佳选择。此时，文档具有“活性”。假如读者的 MATLAB 与 Word 连接正确，用鼠标双击光盘上的 DOC 文件，就能直接进入 Notebook 环境。

（3）光盘 DOC 文件的使用方法

● 作为演练环境使用

在正常打开的光盘 DOC 文件中，读者只要把光标放在绿色的输入细胞内，按组合键[Ctrl＋Enter]，就可使该输入细胞重新执行计算。在演练中，读者可以通过对指令的修改、变化和重新运行，观察运算结果的变化，从而达到举一反三的效果。

● 作为样板使用

先打开光盘 DOC 文件，然后删去原光盘文件内容，再写入读者自己所需的内容，最后通过菜单项的“另存为”操作保存为自己的文件。这样获得的文件能正常地在 Notebook 环境下工作，也就是既可以输入文字、公式，又可以运行 MATLAB 指令、嵌入数值或图形结果；既拥有 Word 的所有文字处理能力，又具备 MATLAB 的运算、表现能力。

B.5 mfiles 文件夹上的 M，MDL 文件的使用

所有算例都以 M 文件或 MDL 文件形式刻录在光盘的\mfiles 文件夹中。

本光盘 M 文件应在 MATLAB R2010a 以上版本运行。

（1）光盘 M，MDL 文件的功用

● 提供可直接运行的 M 源码文件

只要有 MATLAB 环境，本光盘上的 M 文件就可以运行。它的适用条件比\mbook 文件夹上的 DOC 文件宽松得多，也就是说不管读者是否正确安装 Notebook，不管文件产生的是动画还是交互操作界面，它们都能在 MATLAB 环境中正确执行。

每个算例文件都是完整的，可在 MATLAB 环境中直接运行的，所得结果与印刷版相对应。但出于运行方式不同的考虑，有些光盘 M 文件与印刷版文件指令可能会存在少许差别，目的是为把算例特征表现得更充分。

此外，本光盘提供的 M 文件中，有许多是很通用的，读者只要稍加修改，就可为己所用。

● 弥补了印刷版没有 Simulink 模型文件的缺陷

由于 Simulink 工作特点的缘故，所以迄今为止所有涉及 Simulink 的印刷版书籍中都没有能直接运行的模型文件。这给读者带来许多困惑和麻烦：一，读者如想验证书中结论，就不得不从建模做起；二，仿真模块中的参数设置常使初学者顾此失彼，从而造成仿真失败。本光盘的 MDL 模型文件都可直接在 MATLAB 中运行，进行验证。

用户也可以在模型打开后,修改参数,观察变化。

(2) mfiles 文件夹上文件的放置规则

- exm 为前缀的文件都是可直接运行的算例文件

  前缀后的编号与算例编号对应。最左边的两位数字为“章”编号标注,其后两位数字是“节”编号,最后两位数字是节内题号。具体举例如下:

  例【1.3-6】 对应的 M 文件是 exm010306.m ;

  例【7.1-2】 对应的 MDL 模型文件为 exm070102.mdl;

  而配用 M 文件是 exm070102_1.m ;

- 其他非 exm 前缀文件是被调用文件

  在\mfiles 文件夹上还有一些不以 exm 为前缀的文件,它们不与算例直接对应,而是算例必不可少的被调用文件,或是做习题所需的演示文件、数据文件。在印刷版上可以找到有关它们的说明。

(3) M,MDL 的使用方法

直接在 MATLAB 指令窗中,运行(不带扩展名的)算例 M 文件名,就可得到相关结果。在此要再次提醒的是:必须把\mfiles 文件夹设置在 MATLAB 的搜索路径上。

B.6 其他

在本书印刷版发行后,光盘软件的更新内容将通过光盘所带的 Readme 文件作简短的概述性说明。

附录C MATLAB指令索引

C.1 标点及特殊符号指令

C.2 主要函数指令

A a

B b

C c

D d

E e

F f

I i

M m

N n

X x

Y y

Z z

C.3 Simulink 模块

参考文献

[1] 张志涌,等. 精通 MATLAB 6.5 版. 北京:北京航空航天大学出版社, 2003.

[2] MathWorks. MATLAB R2006a. 2006.

[3] MathWorks. MATLAB R2007a. 2007.

[4] MathWorks. MATLAB R2008a. 2008.

[5] MathWorks. MATLAB R2010a. 2010.

［Ｇｅｎｅｒａｌ　Ｉｎｆｏｒｍａｔｉｏｎ］
书名＝ＭＡＴＬＡＢ教程　Ｒ２０１０Ａ
作者＝张志涌，杨祖樱等编著
页数＝３４２
ＳＳ号＝１２６５０６６０
出版日期＝２０１０．０８
出版社＝北京航空航天大学出版社
原书定价＝４３．００
参考文献格式＝张志涌，杨祖樱等编著．ＭＡＴＬＡＢ教程Ｒ２０１０ａ．北京市：北京航空航天大学出版社，２０１０．０８．
内容提要＝本书以ＭＡＴＬＡＢ　Ｒ２０１０ａ为基础修订，系统讲解ＭＡＴＬＡＢ基本环境和操作要旨；分章阐述符号计算、数值计算、计算结果可视化及编程精要；举例展现ＭＡＴＬＡＢ精华工具Ｓｉｍｕｌｉｎｋ的功能级和元器件级仿真能力；举例剖析ＭＡＴＬＡＢ界面编辑器的用法和图形用户界面（ＧＵＩ）的制作要求；简要勾画ＭＡＴＬＡＢ和Ｗｏｒｄ集成一体的Ｎｏｔｅｂｏｏｋ环境。